uc3m | Universidad Carlos III de Madrid

Ingeniería Electrónica Industrial y Automática

2017-2018

Trabajo Fin de Grado

"Desarrollo de una interfaz del terapeuta para la rehabilitación del miembro superior"

Beatriz Martín Calpena

Tutor

Dorin Sabin Copaci

Leganés, 2018

RESUMEN

Este proyecto presenta el desarrollo de una interfaz del terapeuta para el control de un exoesqueleto de rehabilitación del miembro superior. Esta interfaz permite definir con exactitud los parámetros de movimiento de la articulación del codo deseados para una sesión de rehabilitación en función de las necesidades de cada paciente.

La principal función de la interfaz es facilitar el manejo y el control del exoesqueleto, ofreciendo un servicio personalizado y capaz de realizar repetidos movimientos que ayudan a una pronta recuperación motora del miembro afectado. Así mismo, la interfaz está compuesta por elementos que ayudan al control del dispositivo de manera intuitiva y sin necesidad de un usuario con conocimientos sobre programación y electrónica para su uso.

El diseño de la interfaz consiste en un conjunto de objetos que permiten al terapeuta ajustar los parámetros de movimiento tales como las posiciones angulares, la frecuencia y el tipo de movimiento, propios de una sesión de rehabilitación.

Esta interfaz realiza un registro de datos del progreso del paciente, obteniendo los datos más relevantes del movimiento ejecutado para su posterior análisis.

Palabras clave: Exoesqueleto; Terapia de rehabilitación; Registro; Paciente; GUI

ABSTRACT

This project presents the development of a therapist interface for the control of an upper limb rehabilitation exoskeleton. This interface allows defining the movement parameters of the elbow joint desired for a rehabilitation session according to the needs of each patient.

The main function of the interface is to facilitate the handling and control of the exoskeleton, offering a personalized service capable of making repeated movements that help an early motor recovery of the affected member. Likewise, the interface is composed by elements that help control the device intuitively and without the need of a user with knowledge about programming and electronics for its use.

The design of the interface consists of a set of objects that allow the therapist to adjust the movement parameters such as angular positions, frequency and type of movement, typical of a rehabilitation session.

This interface records the progress of the patient, obtaining the most relevant data of the movement performed for further analysis.

Key words: Exoeskeleton; Rehabilitation therapy; Record; Patient; GUI

AGRADECIMIENTOS

Me gustaría agradecer a mi familia y amigos el apoyo y la ayuda durante la elaboración de este Trabajo de Fin de Grado. Asimismo, agradecerles a los profesores, en especial a mi tutor Dorin, la ayuda, orientación y consejo durante mis estudios.

ÍNDICE DE CONTENIDOS

RESUMEN III

ABSTRACT V

AGRADECIMIENTOS VII

ÍNDICE DE FIGURAS XII

ÍNDICE DE TABLAS XV

1. INTRODUCCIÓN Y OBJETIVOS 1

1.1. Robótica de rehabilitación 1

1.2. Interfaces de rehabilitación 1

1.3. Objetivos 2

2. ESTADO DEL ARTE 3

2.1. Exoesqueletos de rehabilitación 3

2.2. Interfaz gráfica GUI 4

2.3. Dispositivos de rehabilitación 5

2.4. Conclusión 18

3. DISEÑO DEL SISTEMA 21

3.1. Exoesqueleto 21

3.2. Electrónica de control 24

3.3. Entorno de programación 25

4. PARÁMETROS DE DISEÑO 27

4.1. Datos a enviar 28

4.2. Datos a recibir 29

5. DESARROLLO DE LA INTERFAZ PROPUESTA 31

5.1. Descripción general 31

5.2. Fases del desarrollo 32

5.3. Funciones de la interfaz 40

5.4. Conexión con el exoesqueleto 50

5.5. Registro de datos..... 51

6. RESULTADOS 53

7. CONCLUSIONES 60

8. TRABAJOS FUTUROS 61

9. MARCO REGULADOR 63

10. PRESUPUESTO 64

11. PLANIFICACIÓN 65

BIBLIOGRAFÍA 66

ANEXO

ÍNDICE DE FIGURAS

Fig. 2.1 Primeros exoesqueletos [3] 3
Fig. 2.2 Interfaz de usuario [8] 4
Fig. 2.3 Alex de Kinetek [9] 6
Fig. 2.4 Ejecución de ejercicios con ALEx [9] 7
Fig. 2.5 Amadeo de Tyromotion [10] 8
Fig. 2.6 Ejercicios terapéuticos para la mano [10] 8
Fig. 2.7 Diego de Tyromotion [11] 9
Fig. 2.8 Armeo Power de Hocoma [12] 10
Fig. 2.9 Paciente utilizando Armeo Power [13] 11
Fig. 2.10 Ejemplo de interfaz del terapeuta [12] 12
Fig. 2.11 Exoesqueleto Armeo Spring [14] 13
Fig. 2.12 Ejercicios terapéuticos del Armeo Spring [14] 14
Fig. 2.13 InMotion ARM [16] 14
Fig. 2.14 Interfaz que recoge datos del paciente [16] 15
Fig. 2.15 Exoesqueleto A2 [13] 16
Fig. 2.16 Dispositivo de rehabilitación ReoGo [15] 17
Fig. 2.17 Juego de rehabilitación de ReoGo [15] 17
Fig. 2.18 Registro de datos del paciente [15] 18
Fig. 3.1 Esquema general del sistema 21
Fig. 3.2 Exoesqueleto del miembro superior 22
Fig. 3.3 Posiciones del brazo 23
Fig. 3.4 Microcontrolador STM32F4 [19] 24
Fig. 4.1 Esquema datos para enviar al exoesqueleto 27
Fig. 4.2 Esquema datos a recibir del exoesqueleto 28
Fig. 4.3 Señal de control 30
Fig. 5.1 Partes de la interfaz 32
Fig. 5.2 Diseño interfaz con parámetros de envío 33
Fig. 5.3 Espacio de representación de la onda sinusoidal 34
Fig. 5.4 Espacio de representación del gif de movimiento 34
Fig. 5.5 Administrador de dispositivos 35
Fig. 5.6 Función detección de puerto disponible 36
Fig. 5.7 Función para abrir el puerto 37
Fig. 5.8 Diseño de la interfaz con recepción de datos 39
Fig. 5.9 Sistema de conexión 40
Fig. 5.10 Barras de ajuste de referencia 43
Fig. 5.11 Menú tipo de movimiento 44
Fig. 5.12 Función botón START/STOP 45
Fig. 5.13 Botón START/STOP 46
Fig. 5.14 Función botón Conectar 46
Fig. 5.15 Botón conexión 47
Fig. 5.16 GIF de movimiento 48

Fig. 5.17 Onda sinusoidal del movimiento 49
Fig. 5.18 Registro de datos en Excel 52
Fig. 7.1 Interfaz de rehabilitación 53
Fig. 7.2 Secuencia flexión extensión 54
Fig. 7.3 Secuencia flexión 55
Fig. 7.4 Secuencia de extensión 55
Fig. 7.5 Funcionalidad botón STOP 55
Fig. 7.6 Controles del exoesqueleto al establecer la conexión 56
Fig. 7.7 Puntos de estudio en la señal de referencia 57
Fig. 7.8 Registro de sesión de rehabilitación 59
Fig. 9.1 Movimientos de pronación y supinación [21] 61
Fig. 11.1 Diagrama de Gantt 65

ÍNDICE DE TABLAS

TABLA 2.1 DISPOSITIVOS DE REHABILITACIÓN 1 19
TABLA 2.2 DISPOSITIVOS DE REHABILITACIÓN 2 19
TABLA 7.1 CONTROLES DE MOVIMIENTO FLEXIÓN-EXTENSIÓN 58
TABLA 7.2 CONTROLES DE MOVIMIENTO FLEXIÓN 58
TABLA 7.3 CONTROLES DE MOVIMIENTO EXTENSIÓN 58

1. INTRODUCCIÓN Y OBJETIVOS

1.1. Robótica de rehabilitación

La robótica de rehabilitación es un ejemplo de la gran evolución de la tecnología en los últimos años. Es una disciplina donde se aplican conocimientos de dos grandes campos como son la ingeniería y la medicina. En el campo de la medicina, la ingeniería ofrece herramientas de control y percepción que, gracias a la interacción que existe entre el ser humano y la robótica, ayudan a una mejora de calidad de vida de los seres humanos.

El principal objetivo de esta área de la ingeniería es el desarrollo de dispositivos tecnológicos con el fin de ayudar a personas con discapacidades o disfunciones motoras. La carencia de movimiento puede ser debido a una enfermedad o a una lesión que repercute en el movimiento de alguna parte del cuerpo. Muchas de estas lesiones son un importante impedimento para el trabajo o para la vida cotidiana, de manera que estas herramientas tecnológicas pueden ser de gran ayuda para fortalecer o recuperar los músculos de la parte del cuerpo afectada.

La ingeniería de rehabilitación también incluye desarrollos de herramientas no sólo con el fin de eliminar las lesiones presentes, sino también para prevenir lesiones que puedan estar en riesgo de producirse en un futuro. [1] Por ello, se diseñan entre otros, dispositivos de rehabilitación para las extremidades superiores e inferiores. En estos diseños, el robot toma una forma muy parecida a la extremidad del ser humano, contando con varias articulaciones y grados de libertad, los cuales coinciden con las de nuestro cuerpo.

1.2. Interfaces de rehabilitación

Gracias al gran avance de la robótica y la programación, se diseñan interfaces con las que el ser humano puede interactuar con el dispositivo de rehabilitación. Estas interfaces mejoran el uso de los robots de rehabilitación y facilitan, tanto al terapeuta como al usuario, la ejecución de la sesión de rehabilitación que se requiere.

Las interfaces humano – exoesqueleto han cobrado gran importancia a la hora de ofrecer una mayor calidad de rehabilitación. Se han convertido poco a poco en algo casi imprescindible a la hora de controlar y medir, de una forma más cómoda, el uso de estos dispositivos tecnológicos.

Además de motivar al paciente por ser capaz de comprobar su progresión a tiempo real, entre sus objetivos destaca el poder reducir el coste hospitalario y el tiempo en el que el personal sanitario o responsable de la sesión de rehabilitación debe estar presente durante el ejercicio. Con esto, las interfaces ofrecen una mayor autonomía al paciente además de aumentar la sensación positiva de ser capaz de valerse por sí mismo.

1.3. Objetivos

El objetivo de este Proyecto de Fin de Grado consiste en el desarrollo de una interfaz para que un terapeuta sea capaz de controlar una sesión de rehabilitación con un exoesqueleto para el miembro superior de un paciente.

Para poder comprobar el funcionamiento de esta interfaz, se ha trabajado con un exoesqueleto de rehabilitación del miembro superior con el que podremos medir y analizar el movimiento de flexión y extensión de la articulación del codo del paciente. Este exoesqueleto podrá ser controlado en todo momento a través de la interfaz, permitiendo ver y ajustar las características del movimiento.

El diseño de esta interfaz tiene como objetivo facilitar el control del exoesqueleto, permitiendo programarlo según el tipo de ejercicio que necesite el paciente en ese momento y sin necesidad de un ingeniero con conocimientos sobre programación para su correcto uso.

Además, esta interfaz tendrá también como objetivo obtener un registro de datos del paciente para poder realizar posteriormente un análisis sobre su evolución por parte del terapeuta.

El desarrollo del proyecto constará de tres fases principales: Diseño de la interfaz, conexión entre la interfaz y un microcontrolador para la validación del control manual, conexión con el exoesqueleto y validación del funcionamiento de la interfaz.

2. ESTADO DEL ARTE

2.1.Exoesqueletos de rehabilitación

El exoesqueleto de rehabilitación es un instrumento electromecánico basado en la estructura y en las funciones motoras que realiza una parte concreta del cuerpo humano. Está diseñado para ayudar a las personas con articulaciones y músculos que no funcionan de manera correcta, ya sea debido a una enfermedad o a una condición física o neurológica. [2]

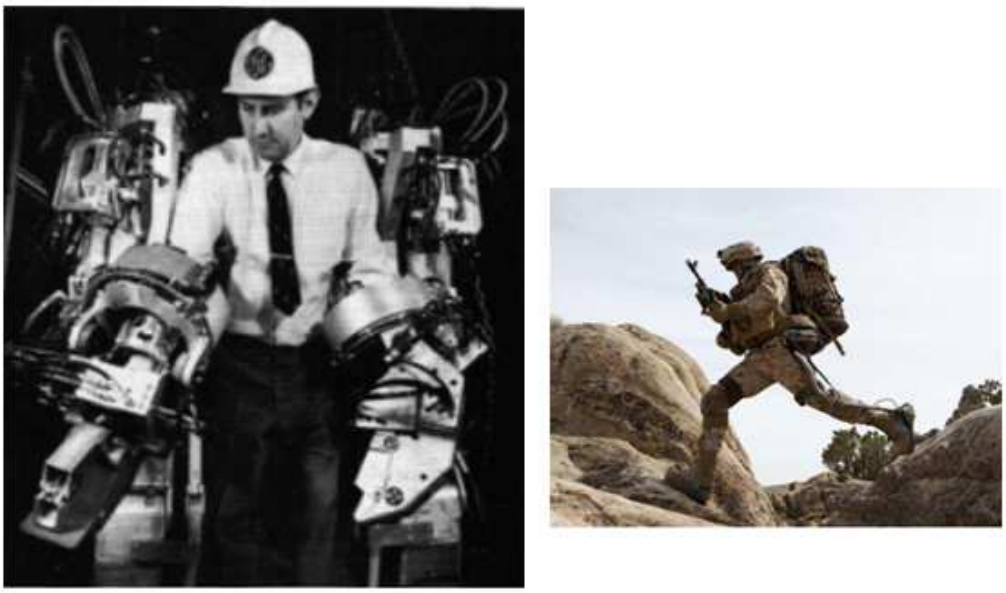


Fig. 2.1 Primeros exoesqueletos **[3]**

Antes de las primeras apariciones de los exoesqueletos de rehabilitación, se diseñaron varios dispositivos robotizados con el fin de facilitar tareas al ser humano.

Tal y como se muestra en la figura 2.1 a la derecha, el primer exoesqueleto comenzó a desarrollarse sobre el 1960 por el ejército de EEUU para aumentar la capacidad motora de los soldados con fines militares. [4]

Algunas compañías también lo construyeron para manejar equipos peligrosos, como la Compañía General Electric, que desarrolló una estructura para manipulación de equipos radiactivos de dos brazos. [5]

Tras la creación de varios exoesqueletos con diferentes funcionalidades como las que se han descrito, La Universidad de John Hopkins en Baltimore, fue de los primeros en diseñar un exoesqueleto para el miembro superior con el fin de ayudar a personas que sufrían de una parálisis motora. [6]

2.2.Interfaz gráfica GUI

Después de algunos años trabajando en el desarrollo de robots y en la informática, al final de los años 70 se comenzaron a desarrollar las primeras interfaces de usuario como en el caso de Xerox, que fue el primero en lanzar una computadora con una interfaz gráfica de usuario. [7]

Tras las primeras creaciones de los exoesqueletos con fines médicos, se comenzaron a desarrollar interfaces de rehabilitación del paciente. Estas interfaces a su vez han ido evolucionando hasta ahora y se han convertido en una ayuda importante en el ámbito de la rehabilitación. Un ejemplo de ello puede verse en la figura 2.2, donde el paciente parece mostrar interés en realizar el ejercicio ayudado de una interfaz.

Fig. 2.2 Interfaz de usuario **[8]**

Una interfaz de usuario (GUI) es un tipo de interfaz que permite establecer una conexión entre usuarios y dispositivos electrónicos, a través de iconos gráficos y otros indicadores visuales. Estos iconos facilitan la interacción entre equipos y personas de manera que no sea necesario utilizar ninguna línea de comandos para manejar un programa o una máquina.

Como cualquier interfaz, aquellas diseñadas específicamente para tareas de rehabilitación con exoesqueletos, han de ser intuitivas, eficientes y diseñadas para poder modificar o manejar, con facilidad, los datos o parámetros que deseemos cambiar entre un movimiento y otro.

Dentro de las diferentes interfaces de rehabilitación podemos distinguir dos tipos según su usuario final. Por un lado contamos con la interfaz del paciente y por otro con la interfaz del terapeuta.

Cada una de ellas está diseñada con una finalidad distinta. La interfaz del paciente tendrá como objetivo motivar y entretener al paciente con ejercicios interactivos, mientras que la interfaz del terapeuta se implementará con fines médicos, siendo capaz de registrar los datos del movimiento y obteniendo los resultados del progreso de su paciente.

2.3.Dispositivos de rehabilitación

La rehabilitación con asistencia robótica persigue el objetivo de optimizar la terapia que necesita el paciente, obteniendo mejores resultados y siempre contando con la revisión del terapeuta o responsable. Estos equipos no son una sustitución de un miembro del cuerpo, sino una ayuda para mejorar el tratamiento de rehabilitación del miembro afectado.

Los dispositivos de rehabilitación están diseñados para poder reproducir los mismos movimientos repetidas veces y con precisión. Con estos dispositivos conseguimos no sólo controlar la fuerza, velocidad y posición del miembro a rehabilitar, sino que además se pueden obtener los resultados del movimiento del paciente y su progresión.

Gracias a la implementación de interfaces, las terapias de rehabilitación son más eficaces, consiguiendo mejorar los resultados y la satisfacción del paciente. Con ellas se consigue proporcionar una mayor motivación al paciente a la hora de realizar los ejercicios correspondientes a su terapia de rehabilitación.

A continuación se describirán algunos de los exoesqueletos de rehabilitación del miembro superior existentes con interfaz de paciente y terapeuta, destacando aquellos cuya principal función se centra en la articulación del codo. Muchos de ellos incluyen módulos de rehabilitación de mano llegando a formar un exoesqueleto más completo de Hombro-Codo-Mano.

- **Alex de Kinetek**

El dispositivo ALEx (*Arm Lightweight Exoskeleton)* es un exoesqueleto robótico fijo diseñado para la rehabilitación de todo el miembro superior del cuerpo humano (figura 2.3). Su diseño proviene de un exoesqueleto previo (PERCRO) de la Escuela Superior de Santa Anna. [9]

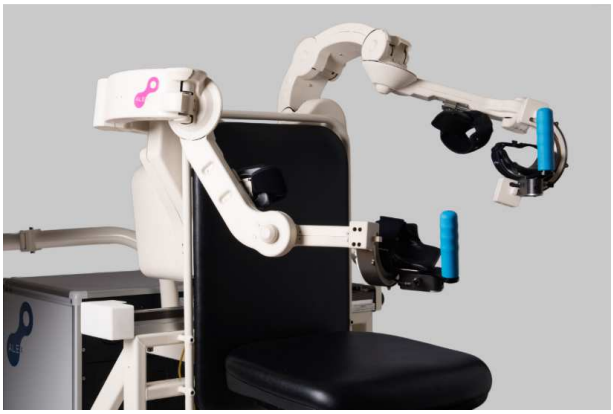


Fig. 2.3 Alex de Kinetek **[9]**

La configuración de doble brazo permite a este robot la capacidad completa de rehabilitación, un tratamiento tanto para el brazo izquierdo como para el brazo derecho del paciente. [9]

ALEx está desarrollado para la rehabilitación clínica para pacientes que hayan sufrido un accidente cerebro-vascular o para la rehabilitación del hombro después de que el paciente de sometiera a una cirugía.

Está fabricado para puedan implementarse interfaces de paciente tales como videojuegos o ejercicios con realidad virtual (figura 2.4). Esto provoca que el paciente reaccione favorablemente a los ejercicios de rehabilitación y por ello, la motivación para ejecutarlos, ya que éstos son totalmente voluntarios. [9]

Fig. 2.4 Ejecución de ejercicios con ALEx **[9]**

Una de las principales ventajas de este brazo robótico es que tiene tres modos de uso para los diferentes movimientos complejos que conlleva el movimiento del brazo: pasivo, activo y asistido. Estos tres tipos se diferencian del grado de autonomía por parte del paciente y se configuran en función de sus necesidades. [9]

- **Amadeo de Tyromotion**

Amadeo (figura 2.5) es un exoesqueleto de rehabilitación para manos y dedos. El dispositivo se puede configurar tanto para niños como adultos.

Este dispositivo es capaz de realizar las tareas de rehabilitación que incluye todas las fases de recuperación de la mano así como analizar su progreso tras la ejecución de estos. [10]

Además, al igual que la mayoría de los exoesqueletos existentes, Amadeo se adapta a las necesidades de cada paciente: tamaño, tipo de rehabilitación, modo de terapia, etc.

Fig. 2.5 Amadeo de Tyromotion **[10]**

Simulando el movimiento natural de la mano, el paciente puede realizar los ejercicios terapéuticos tanto de forma pasiva como activa (figura 2.6). Además, Amadeo da la posibilidad de ejecutar varios tipos de terapias: terapia de asistencia, terapia de movimiento pasivo continuo y terapia inactiva. [10]

Amadeo incluye en sus sesiones tareas interactivas con juegos para motivar la ejecución de los ejercicios de rehabilitación adaptadas al paciente.

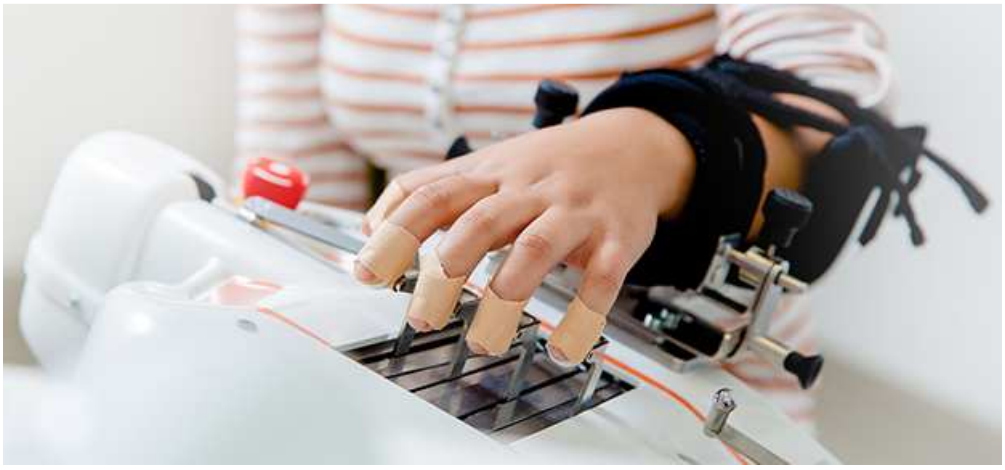


Fig. 2.6 Ejercicios terapéuticos para la mano **[10]**

El software utilizado, tyroS, ofrece un control total sobre todos los procesos de rehabilitación de los dispositivos. Los juegos terapéuticos permiten obtener un feedback a tiempo real y con ello motivar al paciente a continuar con los ejercicios.

Este software recoge los datos terapéuticos relevantes en los registros electrónicos del paciente, representando la evolución gráficamente para una mejor comprensión. [10]

- **Diego de Tyromotion**

Tyromotion ha diseñado más dispositivos de rehabilitación del brazo como es el caso de Diego (figura 2.7), que permite rehabilitar el miembro superior en cualquiera de sus fases.

Al igual que Amadeo, tiene una capacidad de adaptación para todas las edades, permitiendo ajustarlo para cada necesidad del paciente.

Ofreciendo una compensación de la gravedad inteligente asistida por el robot, Diego permite la recuperación motora completa de los brazos, tanto para la articulación del codo como la del hombro.

Debido al diseño colgante mediante cabestrillos, Diego permite una fácil accesibilidad desde cualquier lado del exoesqueleto. Además, este dispositivo ofrece un espacio especialmente adecuado para realizar ejercicios similares a los que el paciente realizaría en su vida diaria con objetos reales, con lo que favorece la motivación del paciente. [11]

Fig. 2.7 Diego de Tyromotion **[11]**

Gracias a sus múltiples aplicaciones para la rehabilitación, Diego se puede adaptar dependiendo del tipo de ejercicio que el terapeuta crea conveniente. Se puede configurar para realizar terapias pasivas, activas y asistenciales, todo ello, aplicando el criterio del terapeuta.

Su diseño de correas con cabestrillos colgantes permite una rápida preparación y por tanto también una rápida configuración de los parámetros del exoesqueleto, El software es el mismo que el exoesqueleto Amadeo (TyroS), lo que facilita la conexión con el dispositivo. [11]

La interfaz de terapia utilizada permite al paciente experimentar mediante realidad virtual tridimensional cada movimiento que realiza el brazo en cada juego.

Esto, junto a la visualización por parte del paciente del registro del progreso, aumenta la motivación para continuar realizando la rehabilitación.

El diseño de la interfaz permite ajustar los ejercicios o juegos en función del nivel de dificultad y duración, y la capacidad del movimiento del paciente.

- **Armeo Power de Hocoma**

El ArmeoPower (figura 2.8) es un exoesqueleto de rehabilitación de hombro y codo que es capaz de combinarse con Manovo de Hocoma, un complemento de rehabilitación opcional para la mano. Dependiendo de la combinación que requiera el paciente, el ArmeoPower puede ser un dispositivo de rehabilitación de hombro-codo o de hombro-codo-mano. [12]

Fig. 2.8 Armeo Power de Hocoma **[12]**

Está diseñado para la terapia de ejercicios en una etapa temprana de rehabilitación.

Este exoesqueleto se ajusta a las necesidades del paciente, ofreciendo más o menos asistencia robótica en función de las capacidades de cada paciente.

Con la ayuda de sensores y algoritmos es capaz de determinar la capacidad de movimiento, ofreciendo mayor soporte a aquellas personas con impedimentos graves de movimiento. [12]

Para aquellas personas con poca movilidad, es posible ajustar unos ejercicios específicos con un alto número de repeticiones para que el paciente recupere las funciones motoras de las extremidades superiores.

Diseñado con seis grados de libertad, ArmeoPower permite al paciente a realizar entrenamientos en 3D para una mejor rehabilitación, aunque también se pueden realizar ejercicios en 1D y 2D. [12]

Para potenciar la motivación y la eficiencia de la terapia, ArmeoPower cuenta con un extenso número de ejercicios o juegos que estimulan al paciente, consiguiendo mejores resultados y un progreso notable (figura 2.9).

Fig. 2.9 Paciente utilizando Armeo Power **[13]**

Estos ejercicios están creados para entrenar la musculatura del miembro superior y ejercitar las funciones motoras que incluyen actividades de la vida cotidiana.

El exoesqueleto registra los datos del paciente y ajusta cada sesión en función de esos datos con ayuda del terapeuta.

Los datos registrados son de gran importancia, pues informan tanto al paciente como al terapeuta de los rangos de movimiento y la fuerza límite del paciente. [12]

En la figura 2.10 se puede ver un ejemplo de los datos que son obtenidos al ejecutar los ejercicios con el Armeo Power, donde puede analizarse la evolución y el movimiento que ha realizado el brazo en todo momento.

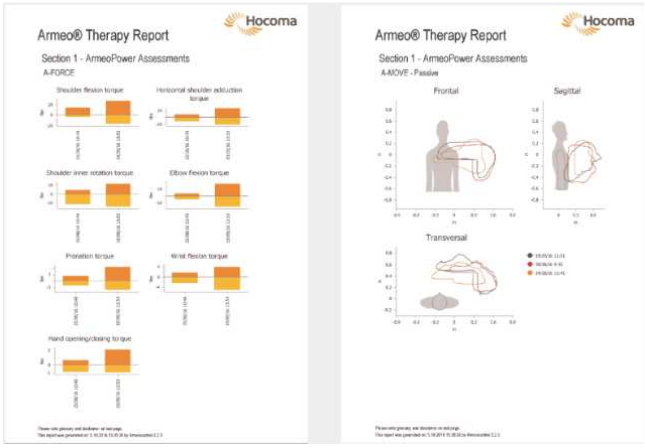


Fig. 2.10 Ejemplo de interfaz del terapeuta **[12]**

El software utilizado permite adaptar el dispositivo a cada juego y paciente, fijando objetivos individuales de terapia. Los juegos se programan con diferentes dificultades poniendo a prueba las funciones motoras.

El feedback después de la terapia será información muy importante, que junto con las nociones médicas, se utilizarán para redactar el diagnóstico y la progresión del paciente. [12]

- **Armeo Spring de Hocoma**

Este exoesqueleto de Hocoma funciona también como soporte para la rehabilitación del brazo. ArmeoSpring (figura 2.11) contrarresta el peso del brazo y permite al paciente centrarse en la repetición de movimientos para la mejora y recuperación de las funciones motoras necesarias.

Los que han colaborado en el diseño de este exoesqueleto aseguran que, despreciando el peso del brazo del paciente y con una buena terapia de repetición de movimientos, se consiguen mejores resultados a largo plazo. [12]

Fig. 2.11 Exoesqueleto Armeo Spring **[14]**

ArmeoSpring está disponible para adultos y para niños *(ArmeoSpring Pediatric)*. Ambos diseños ofrecen una rehabilitación de brazo y mano a la vez, realizando ejercicios terapéuticos mediante juegos.

Los juegos que se utilizan para la terapia de rehabilitación se desarrollan en un espacio en 3D, lo que facilita su realización en el espacio de trabajo. Además admite movimientos en una y dos direcciones, según las necesidades del paciente.

Para un mejor control de la progresión en la terapia, el exoesqueleto registra los datos de los movimientos (rango de movimiento y ángulos de la articulación) durante la sesión de rehabilitación.

Los ejercicios terapéuticos (figura 2.12) forman parte de una biblioteca de ejercicios de realimentación y rendimiento *(Augmented Performance Feedback)*, gracias a los cuales ofrecen un entrenamiento donde se ejecutan movimientos comunes a los de la vida cotidiana. [14, 12]

Fig. 2.12 Ejercicios terapéuticos del Armeo Spring **[14]**

La posibilidad de ver los resultados durante las sesiones aumenta la motivación de los pacientes consiguiendo una mayor independencia en las actividades de la vida diaria. La terapia auto dirigida permite al terapeuta la supervisión de varios pacientes al mismo tiempo sin necesidad de dedicarse únicamente a uno.

El resultado de los datos de cada sesión permite conocer al momento el progreso y el estado del paciente con el fin de mejorar la programación futura de las sesiones del mismo.

- **InMotion ARM de Bionik**

InMotion (figura 2.13) es un exoesqueleto de rehabilitación del miembro superior. Está diseñado para incluir un módulo de terapia de la mano, si fuera necesario, *(InMotion HAND)* que permite el entrenamiento de las funciones motoras de estiramiento y agarre de la mano. [15]

Fig. 2.13 InMotion ARM **[16]**

Los ejercicios del dispositivo están programados para registrar y revisar los parámetros importantes para un buen control de progreso como el rango de movimiento, la fuerza, la velocidad de los movimientos, la coordinación, etc. [16]

Gracias a los sensores y actuadores que forman parte del exoesqueleto, éste es capaz de ofrecer ayuda asistencial en tiempo real, revisando constantemente los parámetros y proporcionando la ayuda necesaria para completar los ejercicios correctamente. Estos parámetros son recogidos para un posterior análisis del progreso del paciente (figura 2.14).

Fig. 2.14 Interfaz que recoge datos del paciente **[16]**

InMotion está preparado para cubrir varias disfunciones motrices tales como la parálisis cerebral, una lesión de la médula espinal o padecimiento de enfermedades como el Parkinson.

La terapia se realiza mediante unos juegos o actividades que invitan a realizar varios tipos de movimiento y así poder controlar el progreso de diferentes parámetros. Algunos de los ejercicios que ponen a prueba la función motora de los pacientes son la repetición de movimientos circulares o lineales. Con ellos se consigue registrar los rangos de coordinación de la musculatura del paciente y el control de cambios de velocidad y precisión del movimiento. [16]

Este dispositivo es capaz de separar el progreso real de la compensación que realiza el robot de manera asistencial. De esta manera tanto el paciente como el terapeuta podrán observar los resultados por separado y comprobar la evolución real del paciente.

- **Nx-a2**

A2 (figura 2.15) es un exoesqueleto de rehabilitación de miembro superior fabricado en Nx, una compañía de China.

Este robot en concreto se basa en una programación dirigida que guía al paciente a realizar los ejercicios correspondientes a su terapia. Estos pacientes sufren de disfunciones motoras debido a enfermedades como apoplejía o hemiplejia y alguna que afectan al sistema nervioso entre otras. [17]

Fig. 2.15 Exoesqueleto A2 **[13]**

Además, gracias a su tecnología virtual integrada en el propio robot, es capaz de proporcionar una valoración de rendimiento y progresión. Esta tecnología se basa en ejercicios interactivos a través de una interfaz que puede enviar mensajes al paciente de motivación o de resultados de forma que aumenta la satisfacción en la terapia.

Mientras el paciente realiza sus ejercicios terapéuticos, el programa es capaz de mantenerle informado de sus progresos, rangos de movimiento, mediante comentarios tanto visuales como audios que motivan e informan también al terapeuta.

Como muchos de los exoesqueletos con interfaces de rehabilitación, también es posible personalizar cada sesión en función de los resultados de otras sesiones y de las indicaciones del terapeuta. Con estos ejercicios se consigue recuperar la actividad muscular, la coordinación y la flexibilidad de las articulaciones del brazo. [17]

Un ejemplo de estos ejercicios interactivos es el juego de la mano de la nube de Tai Chi, donde el robot guía al paciente a realizar movimientos de Tai Chi de manera que le

ofrece una ayuda mental y física. [17] El software de este robot asistencial contiene una base de datos de los pacientes que estén realizando las terapias, registrando todos los resultados durante el ejercicio que se está realizando.

- **ReoGo de Motorika**

ReoGo (figura 2.16) es un robot de rehabilitación diseñado tanto para pacientes adultos como para niños. Su diseño es algo diferente a los que se han descrito anteriormente. Mientras en los anteriores el exoesqueleto y su interfaz se utilizaban sobre una mesa o estructura de soporte fija, ReoGo está diseñado para poder trasladar fácilmente la plataforma con el equipo completo, de manera que facilita notablemente su uso. [15]

Fig. 2.16 Dispositivo de rehabilitación ReoGo **[15]**

El software utilizado combina ejercicios específicos totalmente personalizados a decisión y recomendación del terapeuta. Los ejercicios (figura 2.17) se realizan a través de una interfaz con un monitor donde se reta al paciente interactivamente a ejecutar movimientos necesarios para su recuperación. Además este ofrece opción multilenguaje, lo que amplía el rango de pacientes que puedan utilizar este exoesqueleto. [15]

Fig. 2.17 Juego de rehabilitación de ReoGo **[15]**

Al igual que los anteriores, ReoGo se basa en la repetición de movimientos para fomentar el aprendizaje y activación de la estructura motora del cuerpo. Estas repeticiones son revisadas por el robot proporcionando al paciente una ayuda asistencial con el fin de conseguir los objetivos y una pronta recuperación.

Estos movimientos son continuamente supervisados por el terapeuta y la maquina, quien monitoriza la calidad de los ejercicios en todo momento. Además, cuenta con varias modalidades de uso: guiado, iniciado, paso iniciado y seguimiento asistido. Cada modalidad configura nuevas cargas, rangos de movimiento y demás parámetros para ajustar las necesidades de cada persona. [15]

Al finalizar la terapia, se puede obtener un informe con los progresos y datos de parámetros que el terapeuta deberá tener en cuenta para la próxima sesión de rehabilitación (figura 2.18).

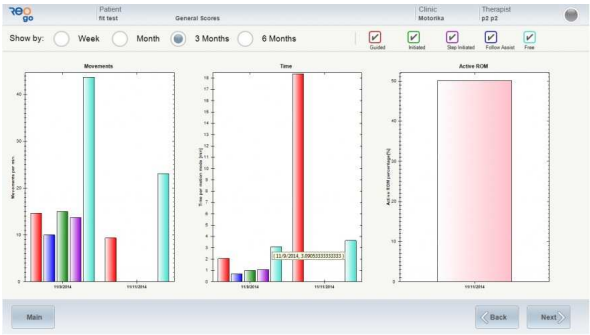


Fig. 2.18 Registro de datos del paciente **[15]**

2.4.Conclusión

Todos estos exoesqueletos, como se ha podido comprobar, tienen como principal terapia la repetición de movimientos. Después de varios estudios [18] y pruebas, se ha demostrado que la repetición de movimientos en terapia favorece el aprendizaje y la recuperación de las funciones motoras. Para que estos movimientos motiven al paciente, se ayudan de juegos o ejercicios cuyo software es capaz de proporcionar un registro de datos relevante para analizar el progreso del paciente.

Aunque la mayoría tengan como objetivo la rehabilitación de la articulación del codo, cada uno tiene un diseño diferente que permite realizar una sesión de rehabilitación

concreta para cada patología y paciente. Muchos de ellos pueden ser ajustables al tamaño de la extremidad superior, siendo válidos incluso para niños.

El continuo estudio por facilitar las sesiones de terapia tanto al paciente como al terapeuta hace que se desarrollen interfaces más complejas en las que se recojan datos a tiempo real y generen un registro que ayuda a comprobar la evolución del paciente.

En las tablas 2.1 y 2.2 se muestra una comparativa de los diferentes dispositivos de rehabilitación presentados anteriormente.

Como se puede observar, todos ellos se ayudan de ejercicios visuales a través de un monitor para mejorar la calidad de la sesión así como ofrecer un informe final de esta para un posterior análisis.

TABLA 2.1 DISPOSITIVOS DE REHABILITACIÓN 1

Dispositivo	Empresa	Adaptación	Rehabilitación	Interfaz	Registro de Datos
ALEx	Kinetek	Izquierdo y derecho	Brazo (hombro y codo)	Videojuegos y realidad virtual	
Amadeo	Tyromotion	Niños y adultos	Manos y dedos	Juegos visuales	Resultados entiempo real (TyroS)
Diego	Tyromotion	Niños y adultos	Brazo y hombro	Juegos vida cotidiana, realidad virtual	Resultados a tiempo real (TyroS)
Armeo Power	Hocoma	Adultos y niños	Brazo + mano	Juegos en 3D, ejercicios terapéuticos	Registro del progreso
Armeo Spring	Hocoma	Adultos y niños	Brazo + mano	Juegos en 3D, ejercicios terapéuticos	Registro del progreso

TABLA 2.2 DISPOSITIVOS DE REHABILITACIÓN 2

Dispositivo	Empresa	Adaptación	Rehabilitación	Interfaz	Registro de Datos
InMotion ARM	Bionik		Brazo + mano	Juegos y actividades	Graficas progreso (fuerza, velocidad, coordinación …)
A2	Nx	Adultos y niños	Brazo	Ejercicios interactivos	Mensajes del progreso en tiempo real
ReoGo	Motorika	Adultos y niños	Brazo	Juegos (Tai Chi)	Informe del progreso

La columna de interfaz correspondería a la interfaz del paciente, mientras que el registro de datos sería la interfaz del terapeuta.

Como resultado del estado del arte presentado en este proyecto, podemos concluir que actualmente, existen varios exoesqueletos que cuentan con interfaz de terapeuta, la cual es imprescindible para un buen control del funcionamiento del exoesqueleto y un seguimiento del paciente.

3. DISEÑO DEL SISTEMA

A continuación, en este capítulo se van a analizar todas las partes que componen el sistema completo del proyecto.

Para crear este sistema es necesario disponer del hardware y software adecuados. Por una parte contamos con el exoesqueleto de rehabilitación, el ordenador utilizado para visualizar la interfaz y trabajar con ella, y un micro controlador. Por la parte del software, se ha hecho uso del programa Matlab para diseñar la interfaz y establecer la conexión con el exoesqueleto. Estos elementos se detallarán más adelante.

En este proyecto podemos distinguir dos partes importantes. Por un lado tenemos el exoesqueleto y por otro la interfaz. Estas dos partes van a estar constantemente conectadas, recibiendo y enviando datos al mismo tiempo de manera que se establece una interacción a tiempo real (figura 3.1).

Fig. 3.1 Esquema general del sistema

A continuación se describen los elementos necesarios para el desarrollo de esta interfaz de rehabilitación que consigue interactuar con el exoesqueleto de rehabilitación.

3.1. Exoesqueleto

El exoesqueleto para la rehabilitación del miembro superior utilizado en este proyecto fue diseñado en la Universidad Carlos III de Madrid (figura 3.2).

Gracias a su ligero diseño, éste puede ser transportado con facilidad, ofreciendo la posibilidad de uso en cualquier lugar siempre y cuando se disponga de una fuente de alimentación. Su colocación e instalación es rápida y sencilla, por lo que es muy cómodo de utilizar.

Este dispositivo se ha diseñado concretamente para la rehabilitación de la articulación del codo y presenta una capacidad de movimiento de hasta dos grados de libertad. Estos dos movimientos corresponden a los de flexión-extensión y pronación-supinación, siendo sólo el primer movimiento el que se use para analizar los datos de progresión del paciente que lo utilice.

Su diseño consta, entre otros elementos, de actuadores SMA (Shape Memory Alloy). Estos actuadores reúnen las características idóneas para su funcionamiento en el exoesqueleto. Cuando un actuador SMA se activa mediante una señal térmica, éste es capaz de encogerse (4% de la longitud total) y después volver a su estado inicial, y así hasta un número elevado de repeticiones.

El exoesqueleto cuenta con sensores de temperatura para medir la temperatura que se alcanza en los actuadores durante su funcionamiento. Además, posee un sensor de efecto Hall que permite señalar la posición del exoesqueleto durante el movimiento de flexión – extensión y un potenciómetro para medir el movimiento de pronación y supinación.

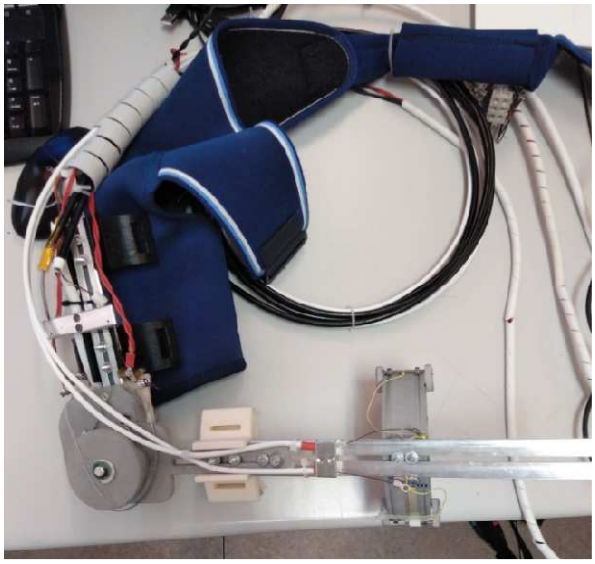


Fig. 3.2 Exoesqueleto del miembro superior

El movimiento de flexión y extensión del codo (figura 3.3) se corresponde con un rango de 0 a 140 grados, similar al del cuerpo humano. Los 0 grados corresponden a la extensión completa del brazo (figura 3.3b) y los 140 grados con la flexión completa del mismo (figura 3.3a). El exoesqueleto cuenta con una máxima frecuencia de movimiento de 0.25 rad/s.

(a) **(b)**

Fig. 3.3 Posiciones del brazo

El exoesqueleto tiene dos puntos de unión con el cuerpo humano: uno en el hombro y otro en la muñeca, pudiendo ajustarse este último en función del paciente.

Este dispositivo de rehabilitación pretende conseguir, mediante movimientos repetidos, reeducar y recuperar las funciones motoras del miembro superior del paciente. Es por ello que los actuadores serán los responsables de ejecutar estos movimientos.

3.2. Electrónica de control

La electrónica del sistema de este proyecto es una parte fundamental del diseño del exoesqueleto para conseguir el control sobre este. Es lo que diferencia una sesión de rehabilitación con un terapeuta a una rehabilitación con un exoesqueleto. Sin la electrónica de control, el exoesqueleto y la interfaz no pueden conectarse e intercambiar datos para su control y sería solo una estructura que necesitaría una fuerza externa para su movimiento.

Como ya se ha explicado anteriormente, el exoesqueleto realiza los movimientos de flexión-extensión gracias a los actuadores (hilos SMA), los cuales se contraen y se estiran en función del movimiento que se le ordene.

A continuación se describen estos elementos necesarios para la electrónica de control.

Tarjeta STM32F4 Discovery

Dentro de la electrónica de control contamos con un microcontrolador STM32F4 de 32 bits (figura 3.5) desarrollado por STMicroelectronics ®. Este microcontrolador ha sido programado con un algoritmo de control para la adquisición de datos con Matlab/Simulink® y así poder simular el funcionamiento que va a tener el exoesqueleto.

Fig. 3.4 Microcontrolador STM32F4 **[19]**

Electrónica de potencia

El control del exoesqueleto posee una electrónica de potencia, donde la señal PWM (Modulación por ancho de pulsos) es generada por el microcontrolador.

El circuito de alimentación de los hilos cuenta con un regulador de tensión para los actuadores SMA, cuya salida servirá para alimentar el circuito de conmutación. Este último circuito consiste en un conjunto de transistores que operan como interruptores de funcionamiento de los actuadores suministrando potencia.

3.3. Entorno de programación

La interfaz va a ser el medio por el cual el usuario se podrá comunicar con el dispositivo de rehabilitación, por tanto, debemos crearla con todas las instrucciones y elementos que se necesiten para el ejercicio de rehabilitación flexión - extensión.

Para programar esta interfaz se ha utilizado la versión R2018a de Matlab®. La elección de este programa se debe a la rapidez del mismo para crear interfaces de una manera clara y visible, además de tener la posibilidad de visualizar gráficos fácilmente que probablemente serían necesarios para el desarrollo del proyecto.

En la interfaz de usuario (GUI) se tienen que ver reflejadas todas las instrucciones de movimiento posibles para el exoesqueleto, además de ejecutar una función u otra dependiendo del ejercicio que se quiera realizar.

Con estas ideas sobre la lógica que debería tener la interfaz, se investigó la forma más completa y resolutiva para crear la interfaz propuesta. El código está diseñado con una programación orientada a objetos, de tal manera que cada elemento de la interfaz corresponde a un objeto. Además, la interfaz consistirá en una programación a eventos, es decir, que la interfaz se va a ir ejecutando cuando haya alguna interacción con ella mediante esos objetos.

La programación orientada a objetos permite separar fácilmente, las acciones que realizan un objeto u otro de la interfaz. Un ejemplo de esto podría ser que al pulsar un botón se encendiese el sistema y al pulsar otro diferente se abra una ventana. Con la

programación a objetos podremos separar las instrucciones que realiza cada objeto pudiendo distinguirlos fácilmente durante la programación y ejecución.

La programación a eventos permite a la interfaz, separar las funciones dependiendo del objeto que se esté utilizando. De esta manera no tendremos un programa principal solo, si no varias funciones que son llamadas al hacer uso de los objetos de la interfaz.

4. PARÁMETROS DE DISEÑO

En este capítulo se va a hacer una breve descripción de los requisitos de diseño de la interfaz propuesta para el control del exoesqueleto de rehabilitación. Se hará una breve introducción a los tipos de datos que se intercambian entre el exoesqueleto y la interfaz.

Como ya se ha explicado anteriormente, la interfaz y el exoesqueleto van a estar conectadas en todo momento y es por eso que, antes de diseñarla, se deben conocer los diferentes datos y parámetros que se reciben y se quieran enviar para el funcionamiento correcto del exoesqueleto.

A continuación, se describirán los diferentes tipos de datos con los que trabajará en la interfaz para la ejecución de los movimientos.

Por un lado detallaremos aquellos datos o parámetros que se tienen que enviar, y por otro los que se debe recibir del exoesqueleto. En las figuras 4.1 y 4.2 se muestran unos esquemas generales de estos tipos de datos, siendo de color azul los que se envían y de color naranja los que se reciben y en color verde el botón de conexión del sistema.

Fig. 4.1 Esquema datos para enviar al exoesqueleto

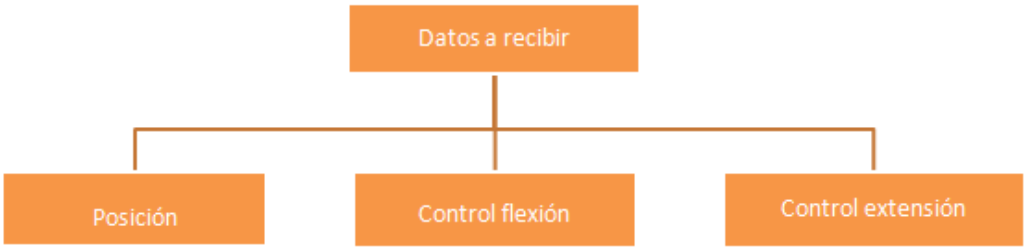


Fig. 4.2 Esquema datos a recibir del exoesqueleto

4.1.Datos a enviar

Como cualquier sistema de control de una máquina o robot, debemos tener en cuenta que, para controlar el exoesqueleto, hay que enviar datos o instrucciones para conseguir el movimiento de este.

Siguiendo los diagramas generales anteriores, a continuación se describen los diferentes datos que se envían al exoesqueleto a través de la interfaz:

- **Referencia**

Estos parámetros son los de referencia de movimiento, es decir, el rango de movimiento que debe tomar el exoesqueleto y la frecuencia de movimiento. Los ángulos de inicio y final se podrán configurar entre unos valores de 0 y 140. Esta referencia corresponde a una señal sinusoidal, donde los ángulos y la frecuencia marcan el trayecto y la velocidad que lleva el exoesqueleto de rehabilitación.

- **Tipo de movimiento**

Aparte de los ángulos de referencia, la interfaz deberá saber qué tipo de movimiento tiene que hacer el exoesqueleto dentro de sus grados de libertad definidos en su diseño. Este movimiento se podrá elegir entre tres diferentes tipos de asistencia: flexión, extensión o flexión – extensión.

- **Botón de STOP**

Este botón puede definirse también como botón de emergencia o botón ON/OFF. Este botón deberá presionarse cuando se quiera comenzar el movimiento (START) una vez se hayan definido los datos de referencia y movimiento, y se pulsará de nuevo cuando se desee pararlo mientras se esté ejecutando el movimiento (STOP).

- **Botón de conexión**

La conexión constante entre el exoesqueleto y la interfaz debe poder controlarse, de modo que no se produzca un envío continuo de datos y se pueda parar el sistema cuando se desee. Por eso, en la interfaz deberá existir un botón que servirá de conexión con el exoesqueleto, de tal manera que al accionarlo se comience con la comunicación e intercambio de datos.

4.2.Datos a recibir

La interfaz no solo debe enviar los datos descritos en el anterior punto si no también recibir otros que certifiquen y señalen el movimiento que está realizando el exoesqueleto.

La interfaz, al igual que tiene que enviar el movimiento que se requiere, también debe comprobar el movimiento que está haciendo realmente. Por ello, la interfaz recoge la posición actual del exoesqueleto y la información de los controles de flexión y extensión.

- **Posición**

Saber la posición que toma en cada momento el exoesqueleto es fundamental para comprobar que este realiza correctamente las funciones que se le indican. Por ello se toman los datos de la posición del codo del dispositivo a través de un sensor de efecto Hall.

Este sensor de efecto Hall señala el ángulo que toma el exoesqueleto en todo momento, tomando valores entre 0 y 140, que es el rango de movimiento flexión-extensión del codo.

- **Control flexión**

El movimiento que realiza el exoesqueleto puede ser debido al movimiento del propio paciente o debido a los actuadores. Si el movimiento es ejecutado por el paciente, las señales de control serán nulas, mientras que si se debe a una orden por parte de la interfaz, comprobaremos sus valores dependiendo del movimiento seleccionado.

Si el movimiento del exoesqueleto se debe a una orden de la interfaz, el parámetro de control de flexión tomará valores entre 0 y 100 dependiendo si está o no sometido a un movimiento de flexión.

Esta señal de control se obtiene a partir de la señal de referencia y la señal que envía el sensor de posición del exoesqueleto, tal y como se muestra en la figura 4.3.

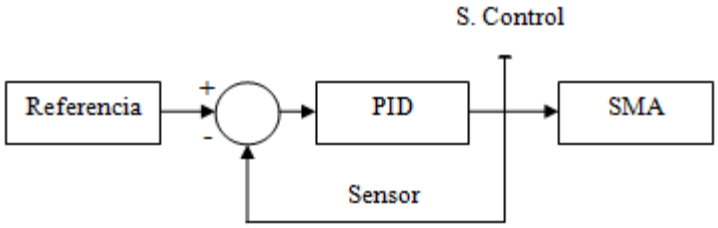


Fig. 4.3 Señal de control

- **Control extensión**

Al igual que el control flexión, el parámetro que recibe de este actuador valdrá para comprobar si el movimiento que se está ejecutando en ese momento es el de la extensión del brazo.

En el caso de elegir el movimiento de flexión, este control tomará el valor 0 mientras que en el de extensión tomará el valores iguales a 100 o cercanos, dependiendo de la señal de referencia (figura 4.3). De la misma manera que con el control de flexión, en el movimiento flexión – extensión variará en función de la dirección del movimiento.

5. DESARROLLO DE LA INTERFAZ PROPUESTA

En este capítulo se hará una descripción del desarrollo de la interfaz del proyecto. En él se incluirá una explicación general, las fases de diseño e implementación y la descripción del código de programación y sus funciones.

Además de todo el desarrollo de la interfaz, también se hará una breve explicación de la conexión con el micro controlador y una pequeña introducción a la funcionalidad del registro de datos.

5.1. Descripción general

Como ya se ha explicado en el entorno de programación, se ha utilizado el programa Matlab para el desarrollo de la interfaz. En este capítulo se entrará un poco más en detalle sobre el desarrollo de la interfaz que se propone en este proyecto.

La interfaz se compone de dos partes bien diferenciadas, la parte visual de la interfaz donde se ve el diseño gráfico, que será lo que utilizará el terapeuta, y el código con las instrucciones del programa.

Estas dos partes estarán conectadas, de tal manera que al hacer uso de uno de los elementos de la interfaz, esta ejecutará una serie de instrucciones definidas en el código. El terapeuta no necesitará tener conocimientos de programación ya que solo necesita la parte visual de la interfaz para trabajar.

No siempre se puede diferenciar al momento el elemento al que nos referimos, ya que en Matlab, hay dos formas de referirse a los objetos dependiendo de donde los utilices.

Un ejemplo de esto podría ser un botón. Cuando queramos configurar el objeto o realizar cualquier acción con él dentro de la interfaz, en el código haremos uso del *"hObject"* si la instrucción se realiza en la misma función que la que se llama al objeto *(Callback)* o usaremos la etiqueta que le da nombre la interfaz cuando queramos realizar alguna acción con él fuera de su función de definición, como por ejemplo *"button"*.

En la figura 5.1 se muestran, de forma general, las dos partes que forman la interfaz.

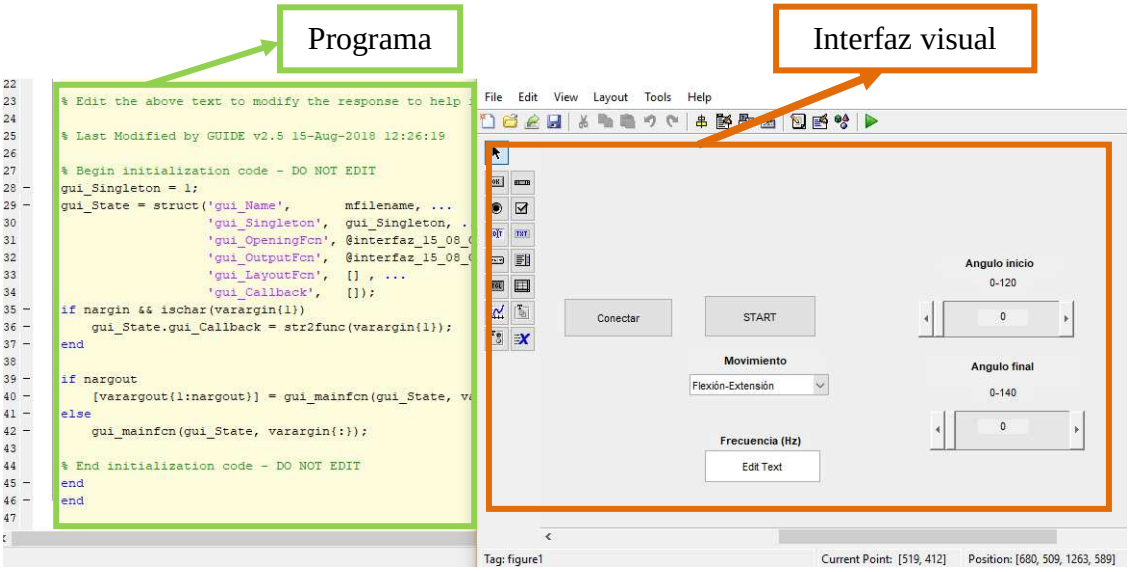


Fig. 5.1 Partes de la interfaz

5.2.Fases del desarrollo

Una vez que tenemos en mente el diseño que queremos, cómo va a ser la programación de la interfaz y los datos que queremos enviar y recibir con ella, comenzamos con el desarrollo de la interfaz.

Para crear esta interfaz de terapeuta se ha ido avanzando por varias fases hasta completar el proyecto. A continuación se describirá en qué consistió cada fase del desarrollo.

Diseño de la interfaz

En esta primera fase se estudió la creación de la interfaz GUI en Matlab con la guía de ayuda de Mathworks [20]. Primero se creó un breve diseño de la parte gráfica de la interfaz.

Para empezar, colocamos el panel de control, donde se implementan todos los elementos con los que interactúa el usuario de la interfaz, que en nuestro caso será el terapeuta.

Una vez definidas las dimensiones del panel, se comenzó a colocar elementos tales como botones, menú de opciones, cuadros de texto para editar, ejes para representar varias gráficas, etc. Todo esto se fue diseñando a partir de los datos de parámetros que la interfaz debía enviar, un ejemplo de algunos de ellos podemos verlo en la figura 5.2.

Para parametrizar la referencia se han usado dos barras móviles con las que poder ajustar los ángulos de inicio y final, seguido de un cuadro de texto para definir la frecuencia de la señal sinusoidal que se envía como referencia de movimiento al exoesqueleto.

El tipo de asistencia se puede elegir con un objeto de tipo menú. Este menú es un desplegable que ofrece las tres opciones de movimiento: flexión, extensión o flexión-extensión.

Para el botón de inicio de movimiento o el botón START / STOP se coloca un objeto de tipo botón.

Fig. 5.2 Diseño interfaz con parámetros de envío

Además de los elementos para definir los datos de envío, se colocaron dos espacios de ejes para representar gráficas, una para mostrar la señal sinusoidal de referencia, y otra para mostrar, mediante una serie de fotogramas, el movimiento del brazo en función del ángulo en el que se encuentre.

Mientras se diseña la interfaz gráfica, se completan las funciones de eventos con instrucciones de configuración o de algún tipo de lógica del programa.

Con esta fase conseguimos un control manual de la interfaz, siendo capaces de momento, de mandar los datos de movimiento desde la propia interfaz para reproducir una gráfica el movimiento del brazo (figura 5.4) y la sinusoidal de referencia (5.3).

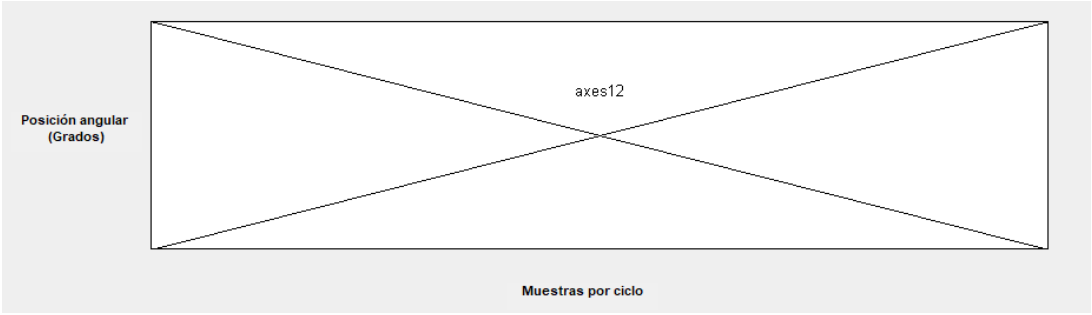


Fig. 5.3 Espacio de representación de la onda sinusoidal

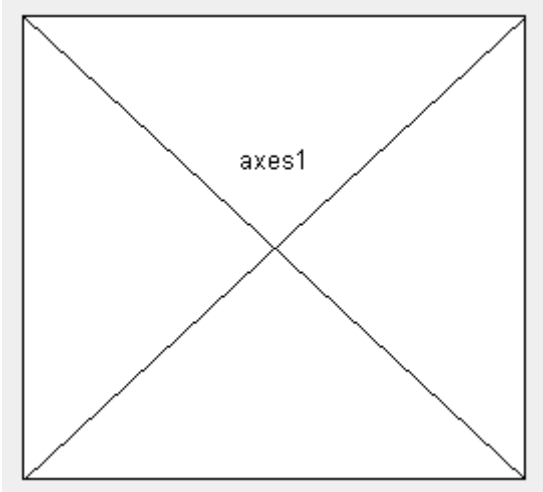


Fig. 5.4 Espacio de representación del gif de movimiento

Habilitar un puerto COM

Una vez diseñada la interfaz con los controles manuales, se procede a la transmisión de datos a través del puerto COM. Para ello hay que configurar una salida para puerto COM en el ordenador que estemos utilizando, detectar el puerto disponible y abrirlo.

Esta fase consta de tres partes, las cuales deben ejecutarse en orden ya que es primordial obtener una buena conexión para la transmisión de datos.

Primero se configurará una salida del PC como puerto COM, luego detectaremos los puertos COM disponibles y por último se inicializará y abrirá el puerto para su posterior uso.

Antes de empezar a trabajar con el puerto COM hay que configurarlo en el PC donde se vaya a controlar la interfaz. Para ello, en administrador de dispositivos (figura 5.5) señalaremos un puerto COM. El puerto seleccionado puede ser tanto de forma automática como manual en caso de que queramos habilitar un puerto específico o cambiarlo.

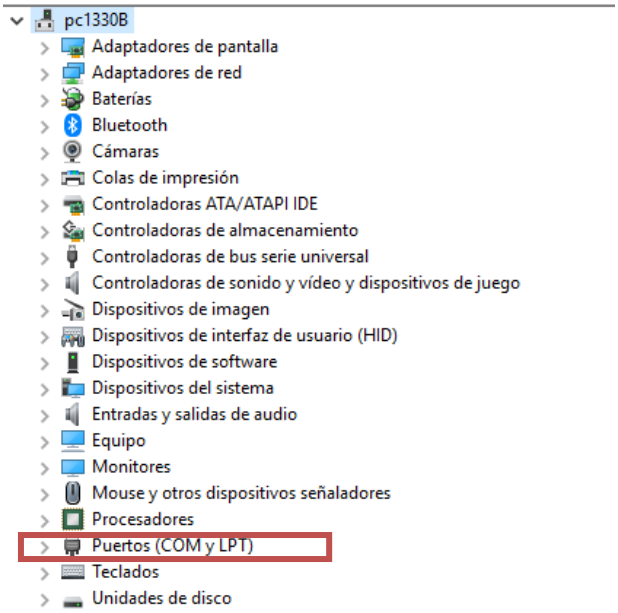


Fig. 5.5 Administrador de dispositivos

Después de habilitar el puerto una vez ya no es necesario repetir este proceso cada vez que se utilice la interfaz. El puerto quedará habilitado permanentemente hasta que se decida cambiarlo por alguna razón.

Ahora se detallarán los pasos previos que deben ejecutarse antes de comenzar la transmisión de datos por el microcontrolador: la detección del puerto disponible y la inicialización de este.

- Detectar el puerto disponible

Después de realizar la configuración para habilitar el puerto COM debemos detectar el puerto disponible. Para ello desarrollamos una función en Matlab que detecte los puertos disponibles en nuestro dispositivo con las instrucciones de la figura 5.6.

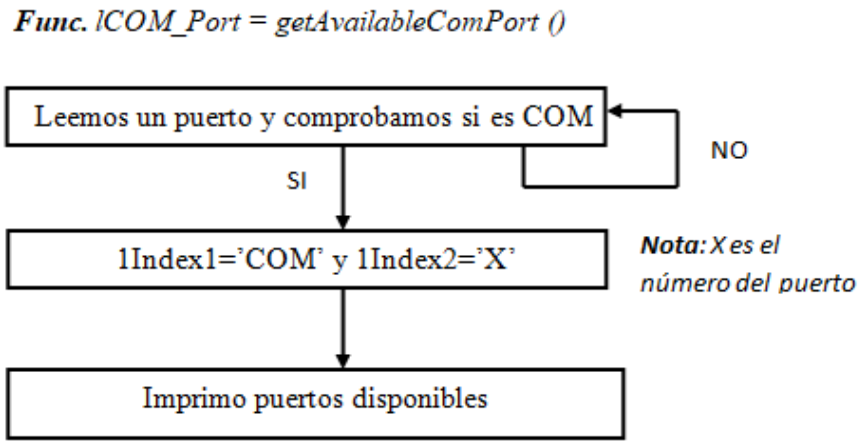


Fig. 5.6 Función detección de puerto disponible

- Abrir el puerto

Cuando sepamos los puertos COM disponibles, se inicializa el puerto serie y se abre para poder realizar la conexión con el micro controlador.

Es importante destacar que, en este proyecto, sin la función para encontrar los puertos disponibles explicada anteriormente, Matlab no podría abrir ningún puerto. Esto es debido a que, según el diseño de la mencionada función, Matlab recoge y guarda en una variable interna el puerto COM seleccionado.

En esta función (figura 5.7), pasamos por parámetro el puerto seleccionado a través de esta variable interna, de tal manera que no es necesario que el usuario lo introduzca manualmente.

Durante todo el proyecto se ha mantenido el objetivo de programar el código de tal manera que, con iniciar la interfaz, todas estas funciones que se mencionan, se ejecuten automáticamente y no sea necesario ningún tipo de conocimiento sobre programación para su uso.

A continuación se muestra en pseudocódigo, la lógica de programación de esta parte importante del programa.

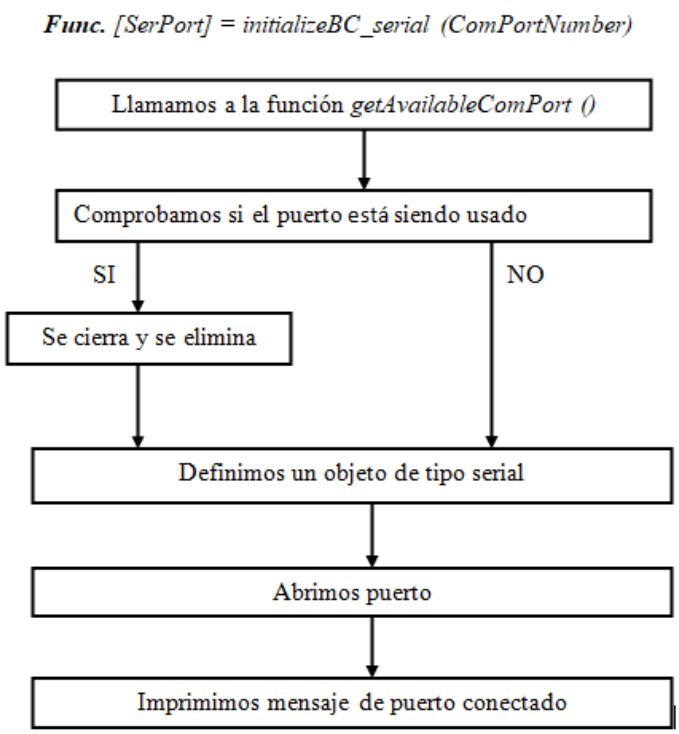


Fig. 5.7 Función para abrir el puerto

Estas dos funciones serán llamadas desde la interfaz cuando se realice la transmisión de datos. Cada una está implementada en dos archivos diferentes con el fin de clarificar las diferentes funcionalidades de cada archivo.

Transmisión de datos manuales

Tras abrir el puerto COM, se procede a la conexión con un micro controlador, el cual está programado mediante la herramienta de Matlab: Simulink para simular el comportamiento del exoesqueleto.

Antes de que la interfaz reciba los datos del exoesqueleto debemos comprobar la transmisión de datos mediante un micro controlador para comprobar la conexión.

Es por esto que se definieron dos funciones para la comunicación, una de ellas para el envío y otra para la recepción de datos. Estas funciones se detallarán más adelante en el apartado de conexiones con el micro controlador.

Las pruebas de conexión comenzaron con valores constantes. De esta manera se comprobó que Matlab era capaz de recibir estos valores y por tanto, si la conexión con el exoesqueleto funcionaría correctamente, pudiendo obtener datos que varían en función del tiempo.

Con todo esto la interfaz ya podía ser controlada manualmente, lo siguiente sería añadir en la interfaz los elementos que indican que la recepción de datos del exoesqueleto y la conexión con este se está efectuando correctamente.

Rediseño de la interfaz

En esta fase se añadieron los elementos necesarios para la conexión del exoesqueleto para poder completar el proyecto propuesto.

Recordamos que los datos que se reciben del exoesqueleto son: el ángulo real del movimiento, el control de la flexión y el de la extensión. Por tanto, en la interfaz, el usuario no debe hacer ninguna interacción con ella para obtener los datos, sino que estos se recibirán automáticamente cuando establezcamos conexión con el exoesqueleto.

En la interfaz se colocan tres bloques de texto editables (Edit text) como en la figura 5.8, los cuales mostrarán los valores del ángulo y el control que esté activo. Esto servirá al terapeuta para comprobar que el exoesqueleto se mueve como se le ha ordenado y que los controles funcionan correctamente.

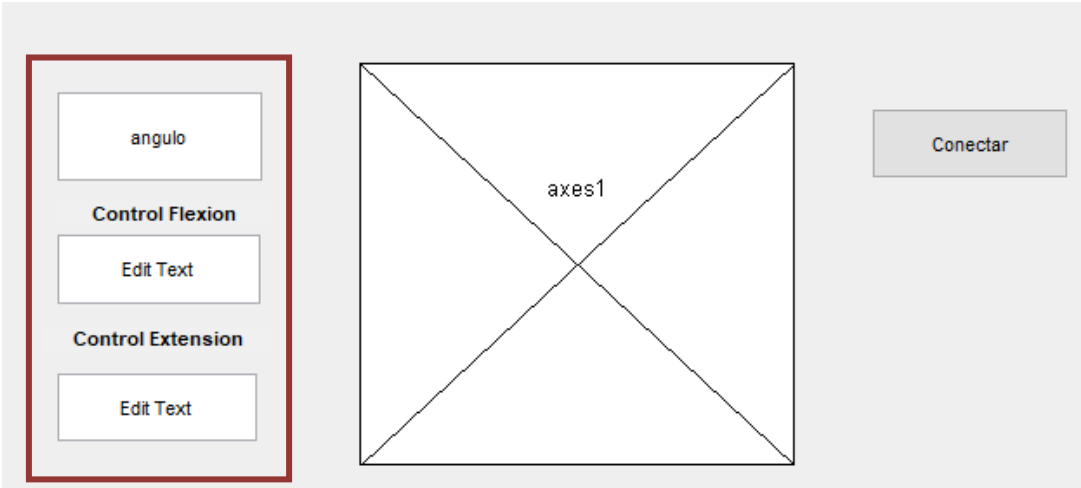


Fig. 5.8 Diseño de la interfaz con recepción de datos

En este punto la interfaz ya estaría completa, tanto en el diseño gráfico como en la implementación del código. A medida que se va diseñando la parte gráfica, se implementa la parte lógica del funcionamiento de la interfaz. En algunos casos la activación de un botón como por ejemplo el START, genera la representación de la señal sinusoidal. Otros en cambio sólo ayudan a parametrizar lo que más tarde ejecutará el movimiento del exoesqueleto.

Conexión con exoesqueleto

Desde el principio se ha trabajado con un micro controlador, programado con la funcionalidad que tiene el exoesqueleto para las primeras fases que incluían alguna modificación en el diseño de la interfaz o algún cambio en la transmisión de datos. Ahora es el turno de conectar el exoesqueleto y la interfaz, es decir, completar el sistema de nuestro proyecto.

En la electrónica de control del sistema, la interfaz y el exoesqueleto identifican las señales de control que se envían y reciben. Una vez conectado el exoesqueleto, podemos hacer las pruebas completas del movimiento de rehabilitación que se desee.

En este proyecto además se implementó una parte de registro de datos. En esta implementación se recogen los parámetros de movimiento y se genera un registro del paciente en una hoja de Microsoft Excel. Con esto el terapeuta puede realizar un análisis del progreso del paciente y obtener los resultados de la sesión de rehabilitación que se requieran.

En la figura 5.9 se muestra la conexión completa del sistema. A la izquierda el exoesqueleto y a la derecha la electrónica de control.

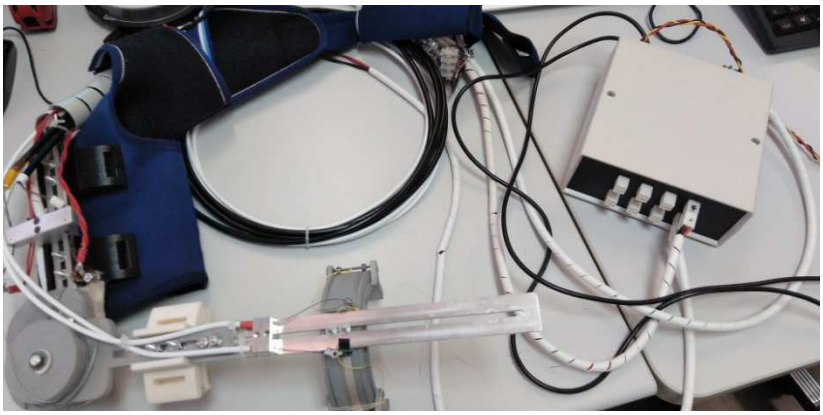


Fig. 5.9 Sistema de conexión

5.3.Funciones de la interfaz

Después de haber visto las fases de creación de la interfaz y su implementación, ahora nos centraremos en algunas funciones que se ejecutan al hacer uso de cualquier elemento de ella.

El terapeuta manejará el exoesqueleto a través de la parte visual de la interfaz por lo que el código de programación será una parte fundamental del proyecto. Esta programación, como ya se ha explicado, consistirá en una programación orientada a objetos y con lógica de eventos.

Al ser una programación a eventos, cada interacción con la interfaz realizará una serie de instrucciones programadas, las cuales determinarán el movimiento del exoesqueleto de rehabilitación. Estas instrucciones están separadas por funciones, lo que corresponde a los eventos o acciones que surgen al hacer uso de la interfaz. A continuación se describirá el programa por funciones o eventos para facilitar la comprensión del funcionamiento de la interfaz.

Las funciones implementadas en la parte del código se pueden clasificar por su funcionalidad dentro de la interfaz.

Por un lado tenemos las funciones de inicio que corresponden a las de configuración e inicialización general de la interfaz. Estas funciones se ejecutan nada más iniciar la interfaz.

Las funciones de objetos serán aquellas que se ejecutan cuando hacemos uso de los elementos de la interfaz, tales como botones, barras, menús, etc.

Por último describiremos las funciones extras. Estas funciones serán aquellas que complementan la funcionalidad y que se ha decidido implementar fuera de las funciones de objetos para clarificar el programa. Estas funciones son las de la reproducción del gif de movimiento y la representación de la señal sinusoidal.

Funciones de inicio

- **Función interfaz()**

Para la inicialización de la interfaz hacemos uso de la función interfaz (varargin). Esta función crea una estructura de tipo GUI. Esta estructura se compone del nombre de la interfaz y de las funciones de apertura y salida de la interfaz, las cuales se explicarán a continuación.

- **Interfaz_15_08_OpeningFcn()**

La función interfaz_15_08_OpeningFcn() es aquella que se ejecuta tras aparecer la interfaz gráfica en el monitor. Esta función no tiene variables de salida, en cambio, posee varios parámetros de entrada como el hObject que se explicó en un capítulo anterior, varargin que permite pasar parámetros desde la línea de comando a la función, y el handles.

El parámetro handles va a estar presente durante todo el programa puesto que es una estructura que contiene todos los objetos de la GUI. Cuando hagamos uso de cualquier elemento durante el programa utilizaremos el handles.

En esta función se inicializan los elementos de la interfaz. Las configuraciones que se establecen son:

- Nombre de la interfaz
- Valores que van a tomar por defecto los ángulos de inicio y fin (0 y 140 respectivamente)
- Inicialización de botones START y el de conexión de datos.
- Valor por defecto de la frecuencia
- Definición de dimensiones de ejes de las gráficas
- Inicialización del gif de movimiento del exoesqueleto

- **Interfaz_15_08_OutputFcn()**

Otra de las funciones que carga la función de inicialización de la interfaz que hemos visto es la función interfaz_15_08_OutputFcn(). Esta función devuelve los parámetros que queremos enviar a la línea de comandos. En nuestro caso la interfaz no debe devolver nada a la línea de comandos por lo tanto devuelve lo que está en la interfaz por defecto.

Funciones de objetos

Las funciones de objetos representan la mayor parte del programa de la interfaz. Estas funciones son las que se ejecutan cuando se ajustan los parámetros de entrada, cuando se inicia movimiento, al iniciar la conexión o al recibir los parámetros de movimiento del exoesqueleto.

Estas funciones se irán detallando en el orden aproximado en el que se usa la interfaz antes de proceder al movimiento del exoesqueleto.

- **Slider1_Callback y slider2_Callback**

Para parametrizar el movimiento hacemos uso de dos barras que se deslizan de 0 a 140. Con estas barras (figura 5.10) ajustaremos los ángulos de inicio y fin del movimiento. Las dos funciones simulan esta asignación de ángulos.

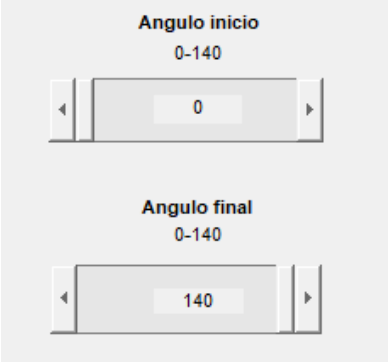


Fig. 5.10 Barras de ajuste de referencia

En ellas se representa en el gif del movimiento la posición que tomaría el brazo en ese ángulo. Con esto el usuario de la interfaz puede comprobar la inclinación del brazo que tomará el paciente si decide ajustarlo a ese ángulo.

- **Popupmenu2_callback**

La función popupmenu2_callback se ejecuta cuando seleccionamos el tipo de movimiento que va a tomar el exoesqueleto de rehabilitación. En este menú aparecerán las siguientes opciones: Flexión-Extensión, flexión y extensión.

Cuando se despliega el menú en la interfaz gráfica (figura 5.11) y se selecciona la opción, esta función obtiene un número que corresponde con el tipo de movimiento elegido.

- ❖ Flexión – Extensión → 1
- ❖ Flexión →2
- ❖ Extensión →3

Para ajustar este valor obtenido de la interfaz al dato de movimiento que se debe enviar al exoesqueleto, restaremos uno a cada variable que corresponde con el tipo de movimiento *tipo_mov*. De esta manera, para los movimientos flexión-extensión, flexión y extensión, los valores que se enviarán realmente serán 0, 1 y 2 respectivamente. Esto es debido a la programación previa del micro controlador, el cual está programado para que funcione con esos valores que indican el tipo de movimiento.

Una vez definidos los parámetros de movimiento, el usuario de la interfaz deberá presionar el botón START para que se inicie el movimiento del brazo.

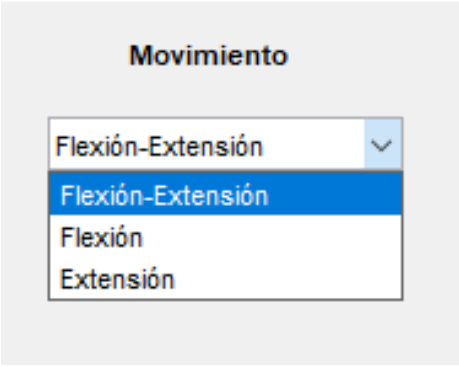


Fig. 5.11 Menú tipo de movimiento

- **Togglebutton4_Callback()**

Cuando se pulsa el botón START / STOP (figura 5.13), la interfaz envía la información necesaria del movimiento al exoesqueleto. Esta información es obtenida de las funciones mencionadas anteriormente, las cuales recogían los parámetros de referencia y los guardaba en el handles.

Una vez recogidos los datos, la interfaz mandará ejecutar al exoesqueleto unas acciones u otras dependiendo del tipo de movimiento seleccionado.

En la figura 5.12 se muestra el proceso descrito que sigue la interfaz al pulsar este botón.

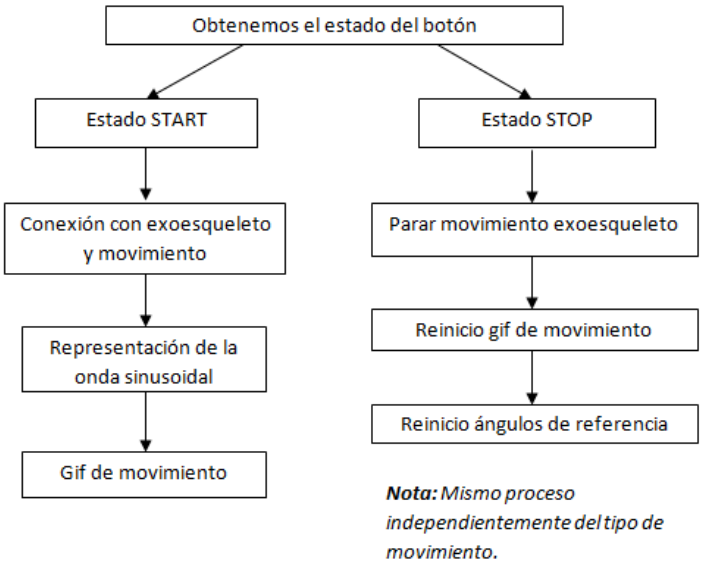


Fig. 5.12 Función botón START/STOP

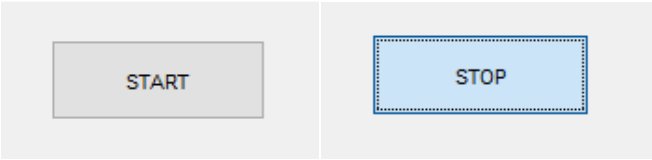
En este botón de la interfaz tenemos dos estados diferentes: el estado 1 corresponde al START y el estado 0 al STOP.

Si el botón se encuentra en START (figura 5.13a), el exoesqueleto iniciará el movimiento que se ha indicado en los parámetros de referencia como los ángulos de inicio y fin, y la frecuencia de movimiento.

Además se representará la función sinusoidal del movimiento que toma el exoesqueleto, así como un gif del recorrido que lleva el mismo.

En el momento que se vuelva a pulsar el botón, que en este caso ya pondrá STOP, todo deberá pararse y reiniciarse todos los ángulos de referencia. El brazo del gif tomará la posición inicial (ángulo 0) y el exoesqueleto deberá dejar de moverse.

Adicionalmente, en el estado de STOP (figura 5.13b), se registrarán los datos del movimiento realizado por el paciente. Esta función de registro de datos se explicará más adelante.



(a) (b)

Fig. 5.13 Botón START/STOP

- **Togglebutton7_Callback()**

Este botón corresponde al botón de conexión adicional que se explicó en el capítulo 4 de este trabajo.

El botón de conexión permite el intercambio de datos entre el exoesqueleto y la interfaz, es decir, determina el estado del puerto COM. En la figura 5.14 se muestra un flujograma que resume el funcionamiento de esta función.

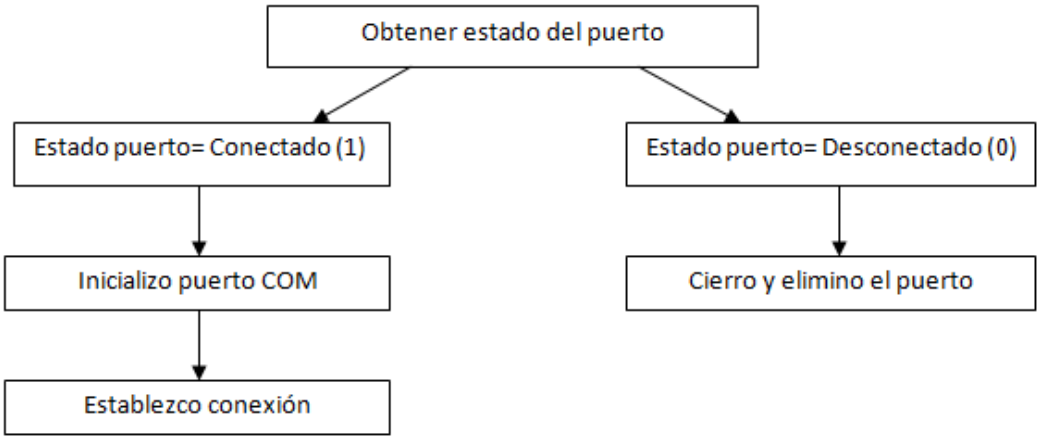
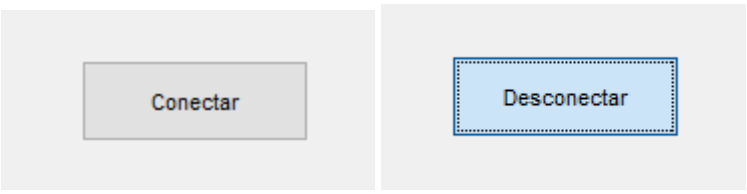


Fig. 5.14 Función botón Conectar

Si se presiona el botón *Conectar* (figura 5.15a) la interfaz recibirá los datos del exoesqueleto. Estos datos son el ángulo de posición y los dos controles de flexión y extensión. Antes de recibir los datos, la interfaz establece la conexión con el puerto COM con todos los pasos para asegurar la correcta conexión.

Si se vuelve a presionar el botón que aparece como *Desconectar* (figura 5.15b), se interrumpe el intercambio de datos a la interfaz y se cierra y elimina el puerto que se esté utilizando para ello.

Estos estados vienen marcados por las etiquetas *Conectar* y *Desconectar* para poder identificar con facilidad las acciones que se realizan al pulsarlo.



(a) (b)

Fig. 5.15 Botón conexión

Funciones extra

Dentro de la interfaz y cuando ha sido necesario, se ha hecho uso de varias funciones para representar el movimiento que toma el exoesqueleto de rehabilitación.

Estas funciones se han implementado sin el uso de ningún elemento de la interfaz, por ello las describiremos como funciones extras. Una de ellas representa el movimiento del brazo mediante un video y otra muestra la onda sinusoidal que realiza el miembro superior mientras se mueve.

- **Display_gif()**

Para que el usuario de la interfaz y el paciente puedan ver con facilidad el movimiento que va a realizar el exoesqueleto, se ha colocado en la parte central de la interfaz un espacio para representar visualmente este movimiento.

Esta representación utiliza un video previamente grabado y divide el video en fotogramas (figura 5.16), de tal manera que cada fotograma representa un ángulo de movimiento, desde el 0 hasta el 140.

Esta representación gráfica se utiliza en algunos puntos de la interfaz como el de visualizar el ángulo de referencia que el usuario está configurando o para ver el movimiento que va a tomar el exoesqueleto cuando se pulsa el botón *START*.

Cada vez que se llame a esta función habrá que especificar el fotograma que queremos que se reproduzca, así como las características del fotograma como por ejemplo el color, tamaño, calidad, etc.

Fig. 5.16 GIF de movimiento

- **Seno2()**

Para representar la onda de movimiento que realiza el paciente cuando está utilizando el dispositivo de rehabilitación también se ha colocado un espacio en la parte superior de la interfaz.

Esta onda con forma de seno viene dada por los ángulos de inicio y fin de referencia, y la frecuencia, ajustada previamente por el usuario de la interfaz.

La gráfica representa el recorrido del brazo del exoesqueleto en función del tiempo donde las cotas superiores e inferiores corresponden a los ángulos de referencia configurados.

En la siguiente fórmula matemática se muestra la obtención de la función sinusoidal:

$$x = A * \sin(2 * pi * f * t) + A + inicio \qquad (1)$$

En ella se hacen uso de parámetros como la frecuencia *f*, el tiempo de muestreo *t* y la amplitud de la onda *A* calculada de la siguiente manera:

$$A = (final - inicio)/2 \qquad (2)$$

Y *final* e *inicio* corresponden a los ángulos de fin e inicio del movimiento respectivamente.

Los valores de la señal se almacenan en una matriz X [1, muestras por ciclo] que se utilizará como parámetro de referencia para enviar los datos al exoesqueleto.

En la figura 5.17 podemos ver representado un ejemplo de la onda sinusoidal calculada por esta función con sus respectivos parámetros (ángulos y frecuencia).

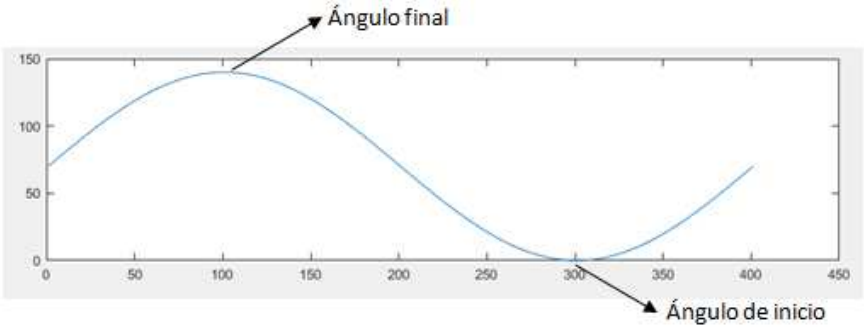


Fig. 5.17 Onda sinusoidal del movimiento

5.4.Conexión con el exoesqueleto

La conexión de la interfaz con el exoesqueleto es una parte imprescindible para la validación del sistema. Sin esta, el sistema no lograría la comunicación entre la interfaz y el exoesqueleto y por tanto no se podría controlar la sesión de rehabilitación.

Esta conexión se realiza mediante el intercambio de datos a través de un micro controlador, que como ya se ha explicado, va conectado a la interfaz por medio de un puerto COM previamente configurado.

En la interfaz se han desarrollado dos canales de comunicación, una para enviar datos y otra para recibir. En las dos funciones se comprueba el estado del puerto, ya que si este no está abierto, no se podrá realizar el intercambio de información entre la interfaz y el exoesqueleto.

El estado del puerto se utilizará como variable de control para intercambiar datos hasta que se interrumpa la conexión o se desconecte mediante el botón habilitado para ello. De esta manera también podemos comprobar que el botón de conexión está funcionando correctamente y que el proceso de inicialización del puerto se ejecuta.

- **Enviar_datos()**

La función *enviar_datos()* es aquella que se encarga de enviar datos al exoesqueleto. Estos datos corresponden a la referencia (onda sinusoidal), el tipo de movimiento y el estado del movimiento.

En esta función se deberán pasar estos datos por parámetro, de tal manera que utilizaremos instrucciones como *fwrite()* para escribir cada dato en el puerto y que pueda ser leído por el exoesqueleto.

Los tres datos que se envían han de ser de tipo single y se envían cada intervalo de 10 ms. De esta forma conseguimos acercarnos lo máximo posible a la conexión a tiempo real.

- **Recibir_datos()**

La función *recibir_datos()* se encarga de recibir los datos del exoesqueleto e introducirlos dentro de la interfaz para poder visualizarlos en ella. Estos datos corresponden al ángulo que toma el brazo, el control de flexión y el control de extensión.

Esta función, al igual que la función de *enviar_datos()*, utiliza el puerto configurado por la interfaz para la transmisión de datos.

Para leer estos datos del micro controlador del exoesqueleto se utiliza la instrucción *fread()* donde se lee, cada intervalo de 10 ms, dato a dato almacenado en el puerto. En nuestro caso serán 3 datos y también de tipo single.

Estos tres datos los asignamos a los elementos de la interfaz encargados de mostrar al usuario de la interfaz mediante la instrucción *set()*. De esta manera conseguimos que los datos leídos se puedan visualizar en la interfaz, mostrándolos en nuestro caso en los espacios asignados para controlar el funcionamiento.

5.5. Registro de datos

Como función adicional al control del exoesqueleto se ha implementado la función ***fichero()*** para obtener un registro de cada sesión de rehabilitación ejecutada.

Este registro permite al terapeuta obtener los datos de la sesión de rehabilitación y la posibilidad de realizar un análisis a partir de ellos. Además, también servirá al paciente para obtener una evaluación de su progresión con las explicaciones médicas del terapeuta.

Los datos se registrarán en una hoja de Excel, donde cada sesión se colocará en cada fila y los datos se repartirán por columnas para una mayor comprensión.

Los datos relevantes que se registrarán serán los correspondientes al movimiento realizado por el exoesqueleto. Estos son el ángulo de inicio y fin del movimiento real, la frecuencia de movimiento y el tipo de movimiento realizado.

En la figura 5.18 se puede observar la distribución de datos de cada sesión de rehabilitación.

Tipo Movimiento	Angulo recorrido	Frecuencia (rad/s)
1	67,171	0,25
2	0,35048	0,25
1	25,66	0,2
3	34,74364	0,25

Fig. 5.18 Registro de datos en Excel

La función fichero consta de dos partes: una primera parte para leer los datos ya existentes y otra para escribir desde el final de estos.

Para leer los datos que ya han sido registrados anteriormente hacemos uso de la instrucción *xlsread()* para leer los datos y *size()* obtener la posición donde termina la lista de datos.

Con esta posición podremos escribir los datos del nuevo registro al final de la hoja. Para escribir los datos nuevos crearemos una matriz de esta manera:

$$A=[dato1, dato2, dato3]$$

Donde dato1 corresponde al tipo de movimiento ejecutado, el dato2 al ángulo recorrido por el exoesqueleto, y el dato3 a la frecuencia de movimiento. Al ser una fila de datos, estos se colocarán automáticamente con la instrucción *xlswrite()* en cada columna.

6. RESULTADOS

En este capítulo se exponen los resultados obtenidos en la elaboración de este trabajo. Por una parte se mostrará el funcionamiento de los controles de la interfaz y por otra el de la conexión con el exoesqueleto.

En la figura 7.1 se muestra el resultado del diseño final de la interfaz. En ella se pueden observar los diferentes elementos que se han ido implementando a lo largo del desarrollo de esta. Por un lado contamos con los controles de parametrización del movimiento y las representaciones visuales del movimiento. Por otro lado en la interfaz también se podrá visualizar los controles del exoesqueleto.

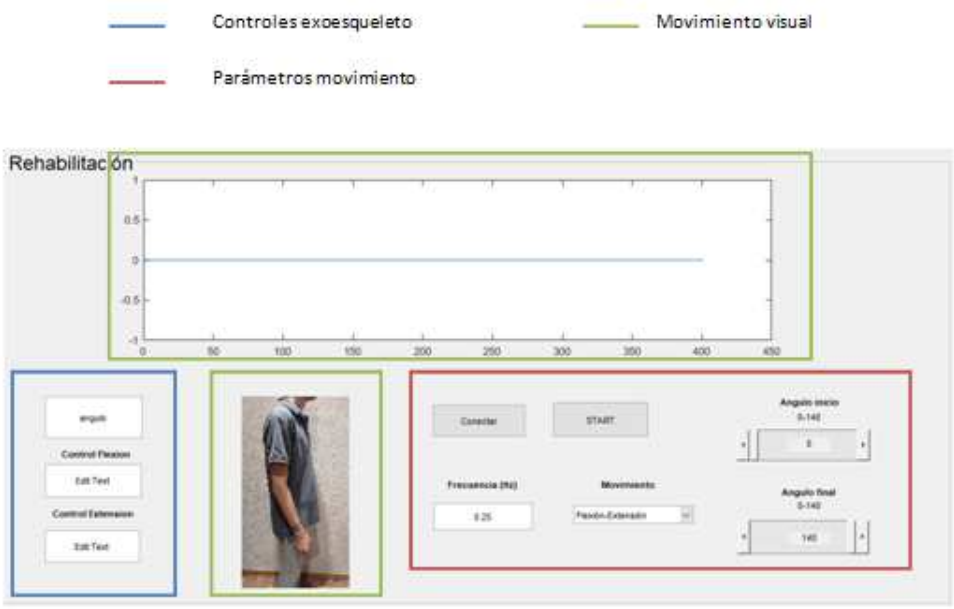


Fig. 6.1 Interfaz de rehabilitación

Antes de comenzar con los controles de la interfaz, se inicia el programa para obtener la ventana de la interfaz sin necesidad de realizar ninguna programación previa. Con el programa principal de la interfaz en Matlab abierto e iniciando el programa (RUN) se obtiene el panel de control de la interfaz tal y como se ha mostrado en la figura 7.1.

Para comprobar todas las funcionalidades que ofrece la interfaz, se hacen dos tipos de ejecuciones: primero se comprobará el control manual para asegurar que se definen bien los parámetros de movimiento, y luego se realizará otra prueba donde se valida el funcionamiento de ésta mientras se ejecuta la transmisión de datos entre la interfaz y el exoesqueleto.

Con la interfaz iniciada y cargada se evalúan los controles manuales. Aquí se comprueba el ajuste del movimiento del exoesqueleto y el funcionamiento del botón de *START/STOP*

El primer paso es definir los parámetros de movimiento (posición angular, frecuencia y tipo de movimiento), teniendo en cuenta que el valor máximo de la frecuencia es 0.25rad/s y 0 y 140 los límites de las posiciones angulares de movimiento (Ver capítulo 5.3). Estos ángulos se pueden configurar deslizando las barras y comprobando en el GIF la posición del brazo que corresponde. La frecuencia se ajusta escribiendo en la casilla correspondiente el valor en rad/s que se desee.

El último parámetro a configurar es el tipo de movimiento, el cual se puede definir con el menú desplegable entre los 3 tipos de asistencia (flexión, extensión o flexión-extensión). Si el usuario no selecciona el tipo de movimiento, la interfaz escogerá el movimiento de flexión – extensión por defecto.

Una vez definidos los parámetros se la sesión de rehabilitación, estos se envían al exoesqueleto mediante el botón *START*. A continuación se inicia la reproducción del GIF que simula el movimiento que va a seguir el paciente con el exoesqueleto.

En las figuras 7.2, 7.3, y 7.4 se muestra por un lado la señal de referencia y por otro una secuencia de imágenes del GIF en función del tipo de movimiento. En la imagen de la derecha se puede ver un ciclo de movimiento de la señal de referencia y en la izquierda la evolución del GIF de movimiento siguiendo los parámetros de movimiento.

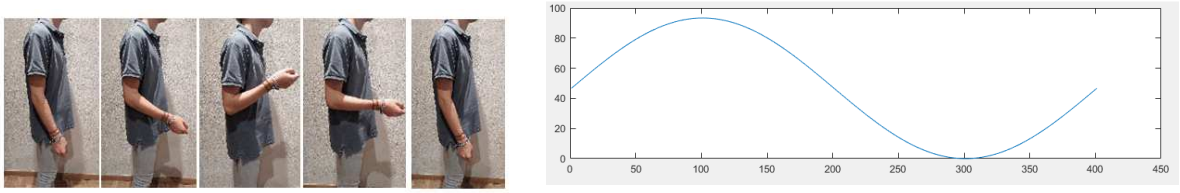


Fig. 6.2 Secuencia flexión extensión

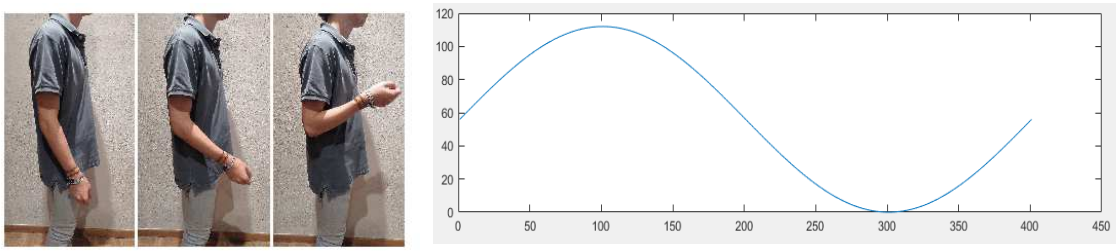


Fig. 6.3 Secuencia flexión

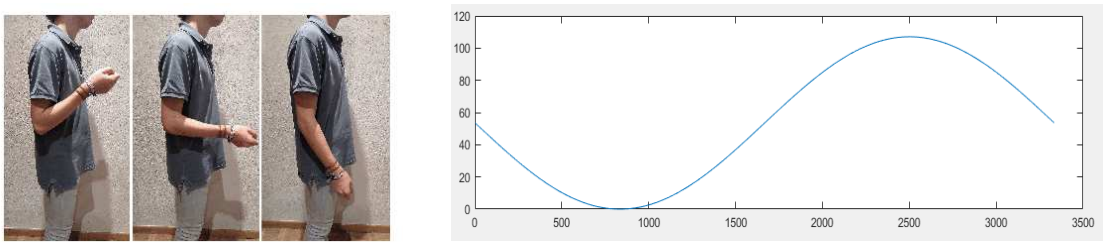


Fig. 6.4 Secuencia de extensión

Con las secuencias de las figuras 7.3 y 7.4 podemos observar las diferencias existentes entre la señal de referencia del movimiento de flexión y el de extensión. Esto confirma que el botón de inicio funciona correctamente enviando los parámetros de movimiento.

Para parar el movimiento una vez iniciado, se presiona el botón *STOP* y observamos que los espacios de ajuste de ángulos se inician a 0 y el botón pasa a llamarse *START*. Además, también reinicia la posición del GIF de movimiento. Estas inicializaciones pueden verse en la figura 7.5.

Fig. 6.5 Funcionalidad botón STOP

Con estos resultados obtenidos se puede afirmar que el control manual de la interfaz funciona de manera adecuada. La interfaz es capaz de recoger los valores de los distintos elementos de la interfaz que definen el movimiento a configurar, y puede reproducir la simulación del movimiento que va a ejecutar el paciente.

Después de la validación de la interfaz manual, se procede a evaluar los controles del exoesqueleto. Estos controles son los que muestran el estado real del exoesqueleto en la interfaz y corresponden al ángulo que toma el brazo del paciente, el control de flexión y el control de extensión.

Con la interfaz iniciada y pulsando el botón de *Conectar*, se comienza con la transmisión de datos. Esta transmisión de datos constará de dos canales de comunicación, uno para recibir datos y otro de envío. La interfaz, tras ejecutar las instrucciones de este botón, muestran en la parte izquierda de la interfaz los valores que está tomando el exoesqueleto en ese momento.

Si nos fijamos en la figura 7.6, en este caso todavía no se ha enviado ninguna instrucción de movimiento por lo que los valores de los actuadores serán 0. En cambio, el primer cuadro de texto mostrará la posición actual del brazo.

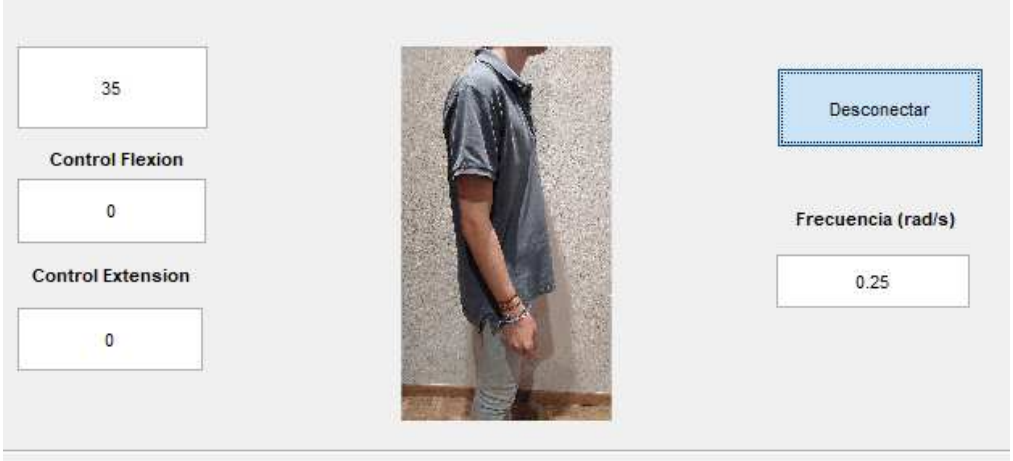


Fig. 6.6 Controles del exoesqueleto al establecer la conexión

Una vez establecida la conexión, se ajusta el movimiento que se requiere en ese momento y se presiona el botón *START* para enviar la ejecución al exoesqueleto.

Cuando se define el movimiento y se mandan esos datos a la interfaz, estamos interrumpiendo el intercambio de datos que se produce cada 10 ms para enviar los nuevos datos de movimiento.

La interfaz posee un tiempo de respuesta mayor, por lo que al interrumpir ese canal de comunicación para establecer los nuevos valores, la interfaz se bloquea impidiendo mostrarlos.

Tras obtener este resultado, se procedió a comprobar si el problema estaba en el canal de comunicación que dejaba de funcionar, o si en cambio, sólo era la interfaz que se bloqueaba al realizar esta interrupción.

Para comprobar el bloqueo que se producía, se realizó la misma prueba con su correspondiente conexión, definición del movimiento e inicio, esta vez mostrando los valores en la pantalla de comandos del propio Matlab. La ventana de comandos mostraba los valores de los controles sin interrumpir el canal de comunicación, por lo que se confirma que la interfaz queda bloqueada debido al rápido intercambio de datos que se produce entre la interfaz y el exoesqueleto y no es un error en el canal de comunicación.

Aunque la interfaz no mostrara los valores en el panel de control de la interfaz, se estudiaron los resultados de 3 casos que validarían la correcta conexión con el exoesqueleto tras definir el movimiento a realizar.

A continuación los datos de los controles que se explicarán en los tres casos serán obtenidos de la ventana de comandos del Matlab. Para ello se tomarán 2 puntos diferentes de la señal de referencia, uno que se dirija a la flexión completa (Punto 1) y otro a la extensión completa (Punto 2) tal y como se muestra en la figura 7.7

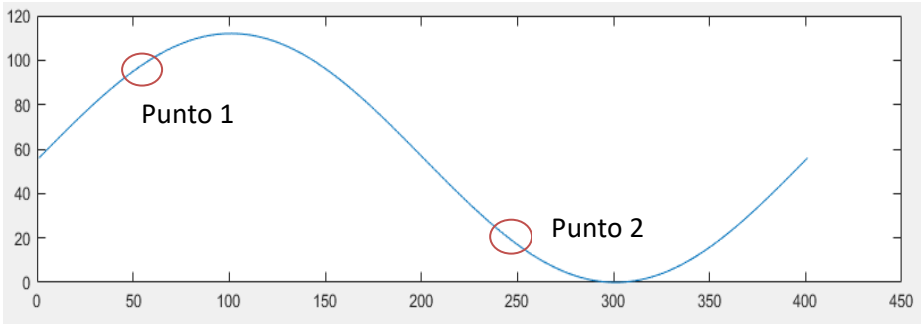


Fig. 6.7 Puntos de estudio en la señal de referencia

Las tablas 7.1, 7.3 y 7.4 representan los valores que toman los controles en función de cada tipo de movimiento y el punto donde se encuentre en la señal de referencia.

- Caso 1: Flexión – extensión

TABLA 6.1 CONTROLES DE MOVIMIENTO FLEXIÓN-EXTENSIÓN

Punto	Angulo	Control flexión	Control extensión
1	100	100 o cercano a él	0 o cercano a él
2	20	0 o cercano a él	100 o cercano a él

- Caso 2: Flexión

Para el movimiento de flexión, el exoesqueleto tomará como valor 0 el control de extensión. Esto es debido a la programación del microcontrolador, que hace que durante este movimiento no se active el control de extensión.

TABLA 6.2 CONTROLES DE MOVIMIENTO FLEXIÓN

Punto	Angulo	Control flexión	Control extensión
1	100	100	0
2	20	0	0

- Caso 3: Extensión

Al igual que en el caso 2, en este caso el control de flexión tomará el valor 0 ya que no se va a realizar ninguna flexión durante este movimiento.

TABLA 6.3 CONTROLES DE MOVIMIENTO EXTENSIÓN

Punto	Angulo	Control flexión	Control extensión
1	100	0	0
2	20	0	100

Registro de datos

La interfaz realiza un registro de datos de movimiento, pudiendo ofrecer la posibilidad de utilizarlo para un posterior análisis del progreso del paciente.

Estos datos se generan automáticamente una vez se inicie el movimiento del exoesqueleto, obteniendo los datos relevantes para el médico especialista encargado. Esta recogida de datos comienza cuando se presiona el botón *START* y se envían los datos al exoesqueleto.

La interfaz genera un archivo Excel automáticamente donde recoge, para cada sesión de movimiento, el tipo de movimiento, la frecuencia y el ángulo recorrido. Estos datos se recogen en una tabla como la de la figura 7.8.

Tipo Movimiento	Angulo recorrido	Frecuencia (rad/s)
1	67,171	0,25
2	0,35048	0,25
1	25,66	0,2
3	34,74364	0,25
1	10,5	0,15
1	35,677	0,25
2	80	0,2
1	58	0,25
3	52	0,15
1	64	0,25
2	90	0,25
1	140	0,25
1	140	0,25

Fig. 6.8 Registro de sesión de rehabilitación

7. CONCLUSIONES

Cumpliendo con los objetivos de este trabajo, se ha desarrollado una interfaz de rehabilitación del miembro superior, donde el terapeuta es capaz de ajustar los parámetros de movimiento de la rehabilitación del codo. Estos parámetros corresponden a las posiciones angulares de referencia, la frecuencia de movimiento y el tipo de asistencia que necesite el paciente, ofreciendo una sesión personalizada. A su vez, esta interfaz permite visualizar el movimiento que realiza el exoesqueleto mediante una secuencia de imágenes, y la señal de referencia que se envía.

A través de la interfaz, el terapeuta puede establecer la conexión con el exoesqueleto y obtener un registro del progreso del paciente en cada sesión, el cual se podrá utilizar para un posterior análisis médico.

La implementación de la interfaz con un solo punto de acceso para su control permite al terapeuta un uso rápido y fácil, evitando el manejo de varios archivos de programación y siendo capaz de ejecutarlos automáticamente desde la interfaz.

Otro de los objetivos perseguidos es este trabajo consistía en un control del exoesqueleto a tiempo real, el cual intercambia datos del movimiento cada 10 ms. Esta transmisión de datos permite conocer la posición que toma el exoesqueleto durante la terapia, así como la dirección de movimiento que sigue (flexión o extensión).

A pesar de no ser un objetivo marcado al principio en este proyecto, la incorporación del botón de conexión y desconexión de datos permite al terapeuta un mayor control sobre la interfaz. Esta comunicación debía ser infinita para obtener la posición real del exoesqueleto, tanto si estaba en movimiento como si estaba parado. Sin este botón, la interfaz entraba en un bucle infinito de transmisión de datos, de tal manera que no se podía interrumpir la conexión en ningún momento. Por tanto, este botón ha añadido funcionalidad a la interfaz.

Con los resultados obtenidos en este proyecto, se puede afirmar que se han cumplido los objetivos fijados en un principio, excepto la visualización en la interfaz de los parámetros del exoesqueleto una vez se envían mediante el botón *START*.

8. TRABAJOS FUTUROS

Tras conocer los resultados y conclusiones de este trabajo, se podría mejorar la comunicación entre la interfaz y el exoesqueleto, de manera que la interfaz posea un tiempo de respuesta menor que el desarrollado y no interrumpa la transmisión de datos entre la interfaz y el exoesqueleto.

Una posible mejora de este sistema de control de rehabilitación podría ser la implementación y el diseño de una interfaz para el paciente. Esta interfaz podría ser un videojuego interactivo a modo de ejercicio terapéutico. Este videojuego estimularía al paciente aumentando la motivación para realizar las sesiones de rehabilitación requeridas.

Después del desarrollo de esta interfaz para la rehabilitación del movimiento de flexión y extensión del codo, también se podría ampliar este control a la medición de los movimientos de pronación y supinación (figura 9.1). El exoesqueleto cuenta en su diseño con este movimiento, por tanto solo haría falta ampliar el diseño de la interfaz y realizar una reprogramación del microcontrolador para la adquisición de datos.

Fig. 8.1 Movimientos de pronación y supinación **[21]**

Por otra parte, el registro de datos y la posterior evaluación juegan un papel importante tras la sesión de rehabilitación. En este aspecto se podría mejorar el análisis de datos incorporando un análisis gráfico, pudiendo visualizar de forma más clara el progreso del paciente por sesión. Además se podrían diferenciar los registros en función del paciente y poder utilizarlo para más pacientes y no solo por un usuario.

Siguiendo con el objetivo principal del trabajo, una posible mejora de la interfaz podría ser, a través de una cámara web, grabar el movimiento que realiza el paciente y que éste lo pueda ver mientras realiza las sesiones de rehabilitación. Con esto también se podrían corregir malas posturas que adquiere el paciente a la hora de realizar las sesiones de rehabilitación. El terapeuta no tiene por qué estar presente durante la sesión, por lo que, con la grabación puede analizar mejor los resultados y evaluar mejor al paciente.

9. MARCO REGULADOR

Sabiendo que existe una normativa para el desarrollo de proyectos de software, en este trabajo no se ha aplicado ninguna norma puesto que el software utilizado ha sido un software libre proporcionado por la Universidad.

En el caso en el que fuera a aplicarse a un proyecto, existen algunos estándares de calidad según la IEEE aplicables.

Normativa IEEE para Software

- *IEEE 1074: Developing Software Lif Cycle Processes*

En esta norma se detallan los procesos del ciclo de vida del Software. [22]

- *IEEE 1219: Software Maintenance*

En esta norma se detallan las actividades para los procesos de mantenimiento y ejecución de software [22]

10.PRESUPUESTO

Descripción	Unidades	Precio/Unidad (€)	Precio total (€)
Exoesqueleto Exoesqueleto portátil de la Universidad Carlos III de Madrid para el miembro superior	**1**	**2000**	**2000**
Ordenador Ordenador portátil Acer Aspire V i3	**1**	**400**	**400**
STM32F4 DISCOVERY Microcontrolador de STMicroelectronics con entradas mini y micro USB	**1**	**17,81**	**17,81**
Cable micro USB Cable USB USB 2.0, 1,2m, Negro, USB A macho a Micro USB B macho	**1**	**5,51**	**5,51**
Cable mini USB Cable USB USB 2.0, 1,8m, Negro, USB A macho a Mini USB macho tipo B	**1**	**3,53**	**3,53**
Licencia Matlab Licencia de estudiante proporcionada por la Universidad	**1**	**0**	**0**
Mano de obra 12 ECTS de matriculación de TFG y 25h/ECTS trabajado	**300**	**15**	**4500**
TOTAL PRESUPUESTO			**6.926,85**

11.PLANIFICACIÓN

En este capítulo se mostrará la planificación que se ha seguido para llevar a cabo el desarrollo del proyecto. Esta planificación se representará mediante un diagrama de Gantt.

En la figura 11.1 se puede observar la representación, por meses, del tiempo dedicado para cada tarea presente en la realización de este proyecto. En él se incluyen las fases del desarrollo de la interfaz, además de tareas como la investigación sobre interfaces en Matlab y la de la elaboración de este documento.

TAREA	Marzo	Abril	Mayo	Junio	Julio	Agosto	Septiembre
Investigación sobre interfaces en Matlab							
Diseño interfaz manual							
Canales de comunicación con el micro							
Rediseño interfaz para ver la conexión							
Conexión con exoesqueleto							
Memoria							

Fig. 11.1 Diagrama de Gantt

BIBLIOGRAFÍA

[1] «National Institute of Biomedical Imaging and Bioengineering,» 2011. [En línea]. Available: https://www.nibib.nih.gov/. [Último acceso: Marzo 2018].

[2] K. Anam y A. Ali Al-Jumaily, «Active Exoskeleton Control Systems: State of the Art,» *Proceding Engineer,* vol. 41, pp. 988-994, 2012.

[3] C. Thompson, «Tech Insider,» 10 Marzo 2016. [En línea]. Available: https://www.businessinsider.com.au/ge-military-robots-2016-3. [Último acceso: Abril 2018].

[4] W. Cloud, « Man amplifiers: Machines that let you carry a ton,» *Popular Science,* vol. 187, nº 5, pp. 70-73&204, 1965.

[5] Mosher, Ralph.S. y S. o. A. Engineers, Handyman to hardiman: Society of Automotive Engineers, New York, 1967.

[6] G. a. W. S. SCHMEISSER, «An upper limb prosthesis-orthosis power and control system with multi-level potential.,» *The Journal of Bone and Joint Surgery (American),* vol. 55, nº 7, pp. 1493-1501, 1973.

[7] I. Villagrán Arroyal, «Retrospectiva de los primeros sistemas de gráficos por ordenador,» *i+Diseño,* vol. 11, nº 11, pp. 34-50, 2016.

[8] «Medical Expo,» [En línea]. Available: http://www.medicalexpo.es/prod/idrogenet/product-74722-829866.html.

[9] Kinetek, «Kinetek,» 2012. [En línea]. Available: http://www.wearable-robotics.com/kinetek/products/alex/. [Último acceso: 14 Mayo 2018].

[10] Tyromotion, «Tryomotion,» 6 Julio 2016. [En línea]. Available: https://tyromotion.com/wp-content/uploads/2016/07/AMADEO_R7_Manual_EN.pdf. [Último acceso: 14 Abril 2018].

[11] Tyromotion, «Tyromotion,» 30 Junio 2015. [En línea]. Available: https://tyromotion.com/wp-content/uploads/2016/03/DIEGO_R1_Manual_EN.pdf. [Último acceso: 25 Mayo 2018].

[12] Hocoma, «Hocoma,» Abril 2015. [En línea]. Available: https://www.stargen-eu.cz/wp-content/uploads/2015/04/Hocoma_Armeo_BRO_Armeo_Therapy_Concept_120420_en.pdf. [Último acceso: 20 Mayo 2018].

[13] «Exoeskeleton report,» [En línea]. Available: https://exoskeletonreport.com/product/armeopower/.

[14] «Hocoma,» [En línea]. Available: https://www.hocoma.com/solutions/armeo-spring/.

[15] Motorika, «Motorika,» [En línea]. Available: http://motorika.com/product-1/.

[16] Bionik, «Bionik,» 17 Agosto 2017. [En línea]. Available: http://eng.interactive-motion.com/usermanual/arm.html. [Último acceso: 23 Abril 2018].

[17] Nx, «Nx,» [En línea]. Available: http://www.yikangshiye.com/?l=en.

[18] R. Cano de la Cuerdaa, A. Molero-Sáncheza, M. Carratalá-Tejadaa, I. Alguacil-Diegoa, F. Molina-Ruedaa, J. Miangolarra-Pagea y D. Torricellid, «Teorías y modelos de control y aprendizaje motor. Aplicaciones clínicas en neurorrehabilitación,» *ELSEVIER,* vol. 30, nº 1, pp. 1-70, 2015.

[19] «Universitat Politècnica de València,» 9 Septiembre 2012. [En línea]. Available: http://www.disca.upv.es/aperles/arm_cortex_m3/curset/guia_iniciacion_STM32F4_discovery.pdf. [Último acceso: 16 Septiembre 2018].

[20] J. Little, M. Cleve y S. Bangert, «Mathworks,» 7 Diciembre 1984. [En línea]. Available: https://es.mathworks.com/discovery/matlab-gui.html?s_tid=srchtitle.

[21] «Periódico de Salud,» 2018. [En línea]. Available: https://periodicosalud.com/supinacion-y-pronacion/. [Último acceso: 16 Septiembre 2018].

[22] «IEEE,» [En línea]. Available: https://www.ieee.org/. [Último acceso: 22 Septiembre 2018].

ANEXO

Interfaz_de_rehabilitación.m

```
function varargout = interfaz_de_rehabilitacion(varargin)
% INTERFAZ_DE_REHABILITACION MATLAB code for
interfaz_de_rehabilitacion.fig
%      INTERFAZ_DE_REHABILITACION, by itself, creates a new
INTERFAZ_DE_REHABILITACION or raises the existing
%      singleton*.
%
%      H = INTERFAZ_DE_REHABILITACION returns the handle to a new
INTERFAZ_DE_REHABILITACION or the handle to
%      the existing singleton*.
%
%
INTERFAZ_DE_REHABILITACION('CALLBACK',hObject,eventData,handles,...)
calls the local
%      function named CALLBACK in INTERFAZ_DE_REHABILITACION.M with
the given input arguments.
%
%      INTERFAZ_DE_REHABILITACION('Property','Value',...) creates a
new INTERFAZ_DE_REHABILITACION or raises the
%      existing singleton*.  Starting from the left, property value
pairs are
%      applied to the GUI before interfaz_de_rehabilitacion_OpeningFcn
gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to
interfaz_de_rehabilitacion_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".

% Inicializacion de la interfaz
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn',
@interfaz_de_rehabilitacion_OpeningFcn, ...
                   'gui_OutputFcn',
@interfaz_de_rehabilitacion_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});

end
```

```
end


%% Funcion de inicio de interfaz
% --- Executes just before interfaz_de_rehabilitacion is made visible.
function interfaz_de_rehabilitacion_OpeningFcn(hObject, eventdata,
handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to interfaz_de_rehabilitacion (see
VARARGIN)

% Choose default command line output for interfaz_de_rehabilitacion
handles.output = hObject;

% Caracteristicas iniciales de interfaz
set(handles.uipanel1,'Title','Rehabilitación');
set(handles.uipanel1,'FontSize', 18);

% Valores por defecto de los angulos
set(handles.text9,'String', 0);
set(handles.slider1, 'Value', 0);
set(handles.text10,'String', 140);
set(handles.slider2, 'Value', 140);

% Valores por defecto del boton de inicio/stop
set(handles.togglebutton8,'String','START');
set(handles.togglebutton8,'Value',0);
set(handles.togglebutton7,'String','Conectar');
set(handles.togglebutton7,'Value',0);


% Valor por defecto de la frecuencia
set(handles.edit4,'String','0.25');

axes(handles.axes12);
global frec
frec=str2double(get(handles.edit4,'String'));

% Valores ejes gif
axes(handles.axes1)
xlim([-2 2])
ylim([-1 1])

% Display gif de movimiento
global imagen
global info
axes(handles.axes1)
imagen=imread('rehab.gif', 'frames', 'all');
info=imfinfo('rehab.gif');

display_gif(imagen,1,info,handles);

% Actualizacion de la estructura handles
guidata(hObject, handles);

end
```

```
%% Funcion de salida de interfaz
% --- Outputs from this function are returned to the command line.
function varargout = interfaz_de_rehabilitacion_OutputFcn(hObject,
eventdata, handles)

% Obtiene la linea de comando de salida por defecto del handles
varargout{1} = handles.output;
end



% Funcion del botón START - STOP
function togglebutton8_Callback(hObject, eventdata, handles)

global a_inicio     %Angulo inicio referencia
global a_final      %Angulo final referencia
global a_int        %Angulo en frames

global imagen
global info
global X

global frec
global n            % Angulo real
global tipo_mov     % Tipo de movimiento
global estado       % Estado del botón
global est_puerto   % Estado del puerto

est_puerto=get(handles.togglebutton7,'Value'); % Estado del puerto
estado=get(hObject,'Value');                   % Estado botón
tipo_mov= get(handles.popupmenu2,'Value');     % Tipo mov seleccionado
tipo_mov=tipo_mov-1;                           % Tipo mov exoesqueleto

%% Display seno
if estado==1 % Si el estado está en START
    set(hObject,'String','STOP');

frec=str2double(get(handles.edit4,'String'));
a_inicio=get(handles.slider1,'Value');
a_final=get(handles.slider2,'Value');

if tipo_mov==0
    axes(handles.axes12);
    X=seno2(frec,a_inicio,a_final);
else
    if tipo_mov==1
        axes(handles.axes12);
        X=seno2(frec,a_inicio,a_final);
    else
        if tipo_mov==2
            axes(handles.axes12);
           X=seno2(frec,a_inicio,a_final);
        end
    end
end



%% Display gif
```

```
axes(handles.axes1);
% Flexion-Extension(bucle flexion y extension - 1 vuelta)
if tipo_mov==0
            for n=a_inicio:a_final


enviar_datos(frec,tipo_mov,estado,est_puerto,X,handles);
                if n <= 4
                    a_int=1;
                else
                    a_int=int16(n/2.5);
                end
                display_gif(imagen,a_int,info,handles);
                pause(0.1);
            end
            for n=a_final:-1:a_inicio
                enviar_datos(frec,tipo_mov,estado,est_puerto,X);
                if n <= 4
                    a_int=1;
                else
                    a_int=int16(n/2.5);
                end
                display_gif(imagen,a_int,info,handles);
                pause (0.1);
            end
else
% Flexion
if tipo_mov==1 && a_final>a_inicio
   for n=a_inicio:a_final
        enviar_datos(frec,tipo_mov,estado,est_puerto,X,handles);
       recibir_datos(est_puerto,handles);
       if n <= 4
            a_int=1;
       else
            a_int=int16(n/2.5);
       end
       display_gif(imagen,a_int,info,handles);
       pause(0.01);
   end
else

% Extension
    if tipo_mov==2 && a_inicio>a_final
        for n=a_inicio:-1:a_final
                pause(0.01)

enviar_datos(frec,tipo_mov,estado,est_puerto,X,handles);
                recibir_datos(est_puerto,handles);
                if n <= 4
                a_int=1;
                    else
                a_int=int16(n/2.5);
                end
                display_gif(imagen,a_int,info,handles)
                pause (0.1);
        end
    end
end
end

else %% (PULSA STOP)
```

```
    % Valores cuando se pulsa STOP
    %datos(n,tipo_mov,estado);
    set(hObject,'String','START');
    set(handles.slider1,'Value',0);
    set(handles.text9,'String', 0);
    set(handles.text10,'String', 0);
    set(handles.slider2,'Value',0);

    c=get(handles.slider1,'Value');
    d=get(handles.slider2,'Value');
    X=seno2(frec,c,d);
    enviar_datos(frec,tipo_mov,estado,est_puerto,X,handles);
    if tipo_mov==1
    a_mov= n-a_inicio;
    fichero(tipo_mov, a_mov, frec);

    else
        if tipo_mov==2
            a_mov= a_inicio-n;
            fichero(tipo_mov, a_mov, frec);
        else
            if tipo_mov==3 % (mirar controles flexion extension)
                a_mov= a_inicio-n;
                fichero(tipo_mov, a_mov, frec);
            end
        end
    end
end
end


%% --- Executes on slider movement.
function slider1_Callback(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%handles.slider1= hObject

global angulo_inicio
global angulo_int
global imagen
global info

angulo_inicio= get(hObject,'Value');
set(handles.text9,'String', round(angulo_inicio,0));

% Display de capturas del gif
if angulo_inicio/2.5 <= 4
    angulo_int=1;
else
angulo_int=int16(angulo_inicio/2.5);
end

display_gif(imagen,angulo_int,info,handles);
end

%% --- Executes during object creation, after setting all properties.
function slider1_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end

%% --- Executes on slider movement.
function slider2_Callback(hObject, eventdata, handles)
% hObject    handle to slider2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global angulo_fin
global angulo_int
global imagen
global info

angulo_fin= get(hObject,'Value');
set(handles.text10,'String', round(angulo_fin,0));

if angulo_fin/2.5 <= 4
    angulo_int=1;
else
angulo_int=int16(angulo_fin/2.5);
end

display_gif(imagen,angulo_int,info,handles);

end

%% --- Executes during object creation, after setting all properties.
function slider2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end

% --- Executes on selection change in popupmenu2.
function popupmenu2_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu2
contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
popupmenu2
get(hObject,'Value')
end
```

```
%% --- Executes during object creation, after setting all properties.
function popupmenu2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

%% --- Executes during object creation, after setting all properties.
function text9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to text9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
end

%% --- Executes during object creation, after setting all properties.
function text10_CreateFcn(hObject, eventdata, handles)
% hObject    handle to text10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
end

%% --- Executes during object creation, after setting all properties.
function axes1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to axes1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: place code in OpeningFcn to populate axes1
end

%% --- Executes during object creation, after setting all properties.
function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
%        str2double(get(hObject,'String')) returns contents of edit4
as a double
end

%% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

%%
function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%        str2double(get(hObject,'String')) returns contents of edit5
as a double
end

%% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit6 as text
%        str2double(get(hObject,'String')) returns contents of edit6
as a double

end

% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
end

% --- Función del botón de conexión.
function togglebutton7_Callback(hObject, eventdata, handles)
% hObject    handle to togglebutton7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of togglebutton7

global serPort
global frec
global tipo_mov
global estado
global X
global est_puerto

est_puerto=get(hObject,'Value');

if(est_puerto==1)
    set(hObject,'String','Desconectar');
    disp('conectado');
    serPort= initializeBC_serial(9);
    recibir_datos(est_puerto,handles);
    %enviar_datos(frec,tipo_mov,estado,est_puerto,X,handles);

else
    set(hObject,'String','Conectar');
    disp('desconectado');
    fclose(serPort);
    delete(serPort);
    set(handles.togglebutton7,'Value',0);
end
end



function edit7_Callback(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit7 as text
%        str2double(get(hObject,'String')) returns contents of edit7
as a double

end

% --- Executes during object creation, after setting all properties.
function edit7_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end
end

function edit8_Callback(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit8 as text
%        str2double(get(hObject,'String')) returns contents of edit8
as a double
end

% --- Executes during object creation, after setting all properties.
function edit8_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

%% Funcion display gif
function display_gif(imagen,angulo_int,info,handles)
value=get(handles.togglebutton8,'Value');
if(value==1)
    imshow(imagen(:,:,1,angulo_int),
info(angulo_int).ColorTable,'Parent',handles.axes1);
else
    imshow(imagen(:,:,1,1),
info(1).ColorTable,'Parent',handles.axes1);
end
end



%% Funcion registro datos en excel
function fichero (inicio, final, frec)

% Leer excel
excel = xlsread('datos.xls');
[m,n] = size(excel);
a=m+2;
num=num2str(a);
fin_hoja=['A',num];

% Escribir excel
dato1=inicio;
dato2=final;
dato3=frec;
A=[dato1, dato2, dato3];
xlswrite('datos.xls',A,'Hoja1',fin_hoja);
a=a+1;
% num=num2str(a)
% fin_hoja=['A',num];
```

```
end

 %% Enviar datos a la tarjeta
function enviar_datos(frec,movimiento, est,puerto,X,handles)

global tipo_mov
global estado
global serPort
global Xpunto
global frecuencia

Xpunto=X;
%global est_puerto
tipo_mov=movimiento;
estado=est;
frecuencia=frec;
size=1/frecuencia*100;
probando=puerto;

while (probando==1)

 for i=1:size

   % Read and show over Matlab console
   fread(serPort, hex2dec('0E'), 'uint8'); %Header Byte 1
   fread(serPort, hex2dec('0E'), 'uint8'); %Header Byte 2
   data1 = fread(serPort, 1, 'single'); %Data
   data2 = fread(serPort, 1, 'single');
   data3 = fread(serPort, 1, 'single');
   %data4 = fread(serPort, 1, 'single');
   fread(serPort, hex2dec('0A'), 'uint8'); %Terminator 1
   fread(serPort, hex2dec('0A'), 'uint8'); %Terminator 2

   %Display received data
   data_uno=data3
   data_dos=data1
   data_tres=data2
   %Write binary data to the controller
   fwrite(serPort,hex2dec('0E'),'uint8');% Header 1
   fwrite(serPort,hex2dec('0E'),'uint8');% Header 2
   fwrite(serPort,[Xpunto(i), tipo_mov, estado]','single');% Data
   fwrite(serPort,hex2dec('0A'),'uint8');% Terminator 1
   fwrite(serPort,hex2dec('0A'),'uint8');% Terminator 2

   %Idle Until next step
   pause(.01);
 end
end
end

%% Recibir datos de la tarjeta
function recibir_datos(estado_puerto,handles)

control=estado_puerto;
global serPort

% data4= posicion encoder
% data5= control flexion
% data6= control extension
```

```
while (control==1)

   fread(serPort, hex2dec('0E'), 'uint8'); %Header Byte 1
   fread(serPort, hex2dec('0E'), 'uint8'); %Header Byte 2
   data4 = fread(serPort, 1, 'single'); %Data
   data5 = fread(serPort, 1, 'single');
   data6 = fread(serPort, 1, 'single');
   %data4 = fread(serPort, 1, 'single');
   fread(serPort, hex2dec('0A'), 'uint8'); %Terminator 1
   fread(serPort, hex2dec('0A'), 'uint8'); %Terminator 2

angulo=data6;
flexion=data4;
extension=data5;

    set(handles.edit7,'String',35);
    set(handles.edit5,'String',flexion);
    set(handles.edit6,'String',extension);

   %Idle Until next step
   pause(.1);
end
end
```

seno2.m

```
function [X]= seno2(frec,a_inicio,a_final)
% En esta función se realiza el cálculo de la onda sinusoidal de
referencia
% Los datos que definen esta onda son la frecuencia y los ángulos

f=frec;
inicio=a_inicio;
final=a_final;

% A es la amplitud de la onda
A=(final-inicio)/2;
% Periodo de un ciclo
tiempo=1/f;
% Numero de muestras de la onda
muestras=round(tiempo*100);
% Matriz donde se guardan los valores de la onda sinusoidal
X=zeros(1,muestras);

i=0;
% t es el tiempo de muestreo de la onda
for t=0:0.01:tiempo
x=A*sin(2*pi*f*t)+A+inicio;
i=i+1;
% Almacenamos los datos en el vector
X(i)=x;
end
% Representamos los datos en una gráfica
ylabel(Posicion);
xlabel(Muestras);
plot(X)

end
```